**Universidade de Aveiro**
**2010**

Departamento de Electrónica, Telecomunicações e Informática

**JOÃO PEDRO SANTIAGO MAIO**

# DESENVOLVIMENTO DE UM DATALOGGER BASEADO EM PENDRIVE USB

**Universidade de Aveiro**
**2010**

Departamento de Electrónica, Telecomunicações e Informática

**JOÃO PEDRO SANTIAGO MAIO**

# DESENVOLVIMENTO DE UM DATALOGGER BASEADO EM PENDRIVE USB

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Doutor Telmo Cunha, Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

Dedico este trabalho à minha família e amigos.

**o júri**

presidente

Prof. Doutor. António Rui Oliveira Silva Borges

Professor associado do Dep. de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

Prof. Doutor. Jorge dos Santos Freitas Oliveira

Professor adjunto do Dep. de Engenharia Electrotécnica da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria

Prof. Doutor. Telmo Reis Cunha

Professor auxiliar do Dep. de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

**agradecimentos**

Durante a realização desta dissertação, várias pessoas estiveram envolvidas. Directa ou indirectamente, a sua influência contribuiu para a evolução e sucesso deste trabalho. A todos eles o meu sincero agradecimento.

Ao meu orientador, Professor Telmo Cunha, pela confiança depositada, pela dedicação e pelo empenho que demonstrou durante este trabalho. A orientação, os erros apontados e os conselhos dados, em tudo proporcionaram uma vontade de fazer sempre algo melhor.

A toda a minha família, pelos constantes estímulos e incansável apoio. Um especial obrigado aos meus pais, por serem as pessoas que são, por todos os conselhos, pela motivação diária que me transmitem, pela preocupação, pela paciência e pelos esforços que fizeram para que eu tivesse a possibilidade de chegar aqui.

Aos meus amigos e a todos aqueles que contribuíram para esta jornada.

Obrigado a todos!

**palavras-chave** Electrónica, DataLogger, micro-controlador, PenDrive USB, comunicação série, sensor.

**resumo** A necessidade de registar dados de vários acontecimentos e recolher medidas efectuadas por sensores é comum a diversas aplicações. De forma a facilitar o seu registo e recolha de dados em áreas como a saúde, indústria, domótica, meteorologia e outras, foram desenvolvidos mecanismos analógicos e, mais recentemente, digitais para suprir essas necessidades, vulgarmente designados por DataLoggers.

Um DataLogger consiste num dispositivo que recebe dados de um ou vários sensores, processa esses dados e finalmente os armazena em formato digital. Este trabalho tem como objectivo o desenvolvimento de um dispositivo genérico de armazenamento de dados que permita uma fácil recolha dos mesmos. Assim, a ligação com os sensores poder ser efectuada com vários protocolos série e durante o processamento dos dados é inserida uma etiqueta temporal relativa à sua recolha, para que os dados possam ser sincronizados com os dados de outros sensores. No final são armazenados numa PenDrive USB genérica.

Embora não seja focado para uma aplicação específica, o DataLogger desenvolvido terá uso imediato no projecto de investigação denominado INSHORE financiado pela FCT.

**keywords** Electronic, DataLogger, micro-controller, PenDrive USB, serial communication, sensor.

**abstract** The need to record data from various events and collect measurements from sensors is common to many applications. To facilitate data recording in areas such as health, industry, home automation, meteorology among others, analog and, more recently, digital devices (generally denominated DataLoggers) have been developed to overcome these needs. A DataLogger is a device that receives data from one or more sensors, processes and, finally, stores it in a digital format. This work aims to the development of a generic device for storing data which is very flexible and of simple operation. Therefore, the connection with the sensors will consider different serial protocols. During the processing stage, a time tag is inserted into the data for time synchronization with other sensors. The data storage is made in a generic USB PenDrive.

Although not focused on a specific application, the developed DataLogger has immediate use in the FCT research project named INSHORE.

# Índice

# Lista de Figuras

# Lista de Tabelas

# Lista de Siglas e Acrónimos

| | |
|---|---|
| µC | *Microcontroller* |
| 1PPS | 1 Pulso Por Segundo |
| ADC | *Analog to Digital Converter* |
| ALU | *Arithmetic Logic Unit* |
| CPU | *Central Processing Unit* |
| DAC | *Digital to Analog Converter* |
| DC | *Direct Current* |
| DIP | *Dual In-line Package* |
| DMA | *Dynamic Memory Access* |
| EEPROM | *Electrical Erasable Programmable Read-Only Memory* |
| EPROM | *Erasable Programmable Read-Only Memory* |
| FCT | Fundação para a Ciência e Tecnologia |
| GPS | *Global Positioning System* |
| GSPS | *Giga Samples Per Second* |

| | |
|---|---|
| I/O | *Input/Output* |
| I²C | *Inter-Integrated Circuit* |
| ICD3 | *In-Circuit Debugger 3* |
| ICSP | *In-Circuit Serial Programming* |
| IDE | *Integrated Development Environment* |
| INSHORE | *Integrated System for High Operational REsolution in Shore Monitoring* |
| LCD | *Liquid Crystal Display* |
| LED | *Light-Emitting Diode* |
| MCU | *Microcontroller Unit* |
| OTP | *One-Time Programmable* |
| PCB | *Printed Circuit Board* |
| PIC | *Programmable Interface Controller* |
| PMP | *Parallel Master Port* |
| PWM | *Pulse With Modulators* |
| RAM | *Random Access Memory* |
| RISC | *Reduced Instructions Set Computer* |
| ROM | *Read Only Memory* |

| | |
|---|---|
| RS-232 | *Recommended Standard 232* |
| RS-485 | *Recommended Standard 485* |
| RTCC | *Real Time Clock and Calendar* |
| SD | *Secure Digital* |
| SFR | *Special Function Registers* |
| SIP | *Single In-line Package* |
| SMD | *Surface Mount Device* |
| SOIC | *Small Outline Integrated Circuit* |
| SOT | *Small-Outline Transistor* |
| SPI | *Serial Peripheral Interface Bus* |
| UART | *Universal Asynchronous Receiver Trasmitter* |
| USB | *Universal Serial Bus* |
| TTL | *Transistor–Transistor Logic* |
| WDT | *WatchDog Timer* |
| XTAL | *Crystal Oscillator* |

# Capítulo 1

# 1 Introdução

Este trabalho incide sobre o desenvolvimento de um sistema de armazenamento de dados provenientes de sensores. Este tipo de sistemas é necessário em áreas como a indústria, domótica, navegação, meteorologia, entre outras. Com a evolução da tecnologia que se tem sentido nos últimos tempos, a necessidade de adquirir dados sobre vários tipos de acontecimentos, processos e eventos específicos tem aumentado consideravelmente. No entanto, a disparidade de aplicações é tal que, factores como capacidade de armazenamento, volume do dispositivo, sincronização entre vários sensores a funcionar em simultâneo, interface de descarregamento de dados, portabilidade, entre outros, tornam estes sistemas de armazenamento demasiado divergentes. O trabalho aqui apresentado considera, então, a concepção e desenvolvimento de um dispositivo de armazenamento de dados que seja versátil e facilmente configurável, e que permita efectuar a afectação de etiquetas temporais sincronizadas com um relógio universal (tempo GPS – *Global Positioning System* – fornecido por um receptor GPS).

Neste Capítulo é feito um breve enquadramento sobre o trabalho a realizar, seguindo-se os objectivos técnicos pretendidos a considerar durante o decorrer do trabalho e, por fim, é apresentada a estrutura geral desta dissertação.

## 1.1 Enquadramento

Em aplicações como saúde, segurança de edifícios e automóveis, sistemas de navegação integrando vários sensores de localização, processos fabris, estações meteorológicas, monitorização ambiental, e muitos outros, é indispensável

armazenar dados provenientes de sensores. Dependendo do tipo de sensores, os dados a armazenar podem ter características muito distintas, nomeadamente na característica temporal que apresentam. Assim, os dados podem ser enviados pelos sensores muito esporadicamente (tipicamente em aplicações de monitorização com horizontes temporais longos), periodicamente em pacotes de dados concentrados no tempo (como é o caso dos sensores de sistemas de navegação, tais como os receptores GPS), ou de forma mais ou menos contínua. Contudo, é uma característica comum à maior parte dos sensores considerados que a transmissão que estes apresentam se baseia num protocolo série (RS232, $I^2C$, etc.). Assim, surge a necessidade de se criar um dispositivo que seja simples e de baixo custo que possa ser acoplado ao sensor e que fique encarregue por armazenar os dados que o sensor envia. Esse dispositivo, aqui denominado de DataLogger, deve ser facilmente configurável para se adaptar facilmente às características de cada sensor específico. Adicionalmente, é conveniente que os dados acumulados no DataLogger sejam transferidos para um computador (que irá, posteriormente, analisá-los) de uma forma simples e pouco morosa. Verifica-se que a maioria dos dispositivos existentes no mercado para efectuar o armazenamento genérico de dados não apresenta um leque de possibilidades de configuração suficientemente largo, assim como não oferece um meio expedito para transferir esses dados armazenados para um computador. Também se verificou não existir no mercado um dispositivo de armazenamento que registasse o instante de chegada de conjuntos de dados com referência a um referencial de tempo universal (como o tempo GPS) por forma a se poder adquirir dados de vários sensores de uma forma síncrona (i.e., com vários dispositivos de armazenamentos ligados a sensores dispersos numa região, efectuar o registo dos seus dados em que estes são afectos a uma referência temporal única, podendo assim processar os dados de uma forma conjunta pois eles se encontram sincronizados).

As lacunas acima referidas nos dispositivos de armazenamento de dados comerciais motivaram o desenvolvimento de um novo dispositivo – o DataLogger

– que possa colmatar essas deficiências. Mais ainda, no âmbito do projecto de investigação INSHORE - *Integrated System for High Operational REsolution in Shore Monitoring* – projecto financiado pela Fundação para a Ciência e a Tecnologia e que decorre no Instituto de Telecomunicações, é pretendido efectuar a recolha síncrona de sensores de navegação (uma unidade de sensores inerciais – acelerómetros e giroscópios – e um distanciómetro laser de precisão) de uma forma síncrona, simples e flexível, pelo que este trabalho também foi motivado pelos objectivos deste projecto de investigação.

Com o progresso tecnológico nos últimos tempos, tem-se notado um desenvolvimento muito acentuado nos micro-controladores, o que não só proporciona um considerável aumento na quantidade de recursos e da própria fiabilidade, mas também uma considerável diminuição de custo. Acrescentando ainda a sua alta mobilidade e autonomia, os micro-controladores tornam-se os componentes ideais para a base de desenvolvimento de um sistema de armazenamento de dados.

## 1.2 Objectivos

Com a crescente proliferação dos sistemas digitais pelas mais diversas áreas, são várias as aplicações onde é necessário armazenar dados provenientes de sensores. Numa grande quantidade dos casos, pretende-se que os dados gerados por um conjunto de sensores sejam armazenados durante extensos períodos de tempo, gerando assim volumes de dados de elevada dimensão. Dados esses que *a posteriori* serão analisados em gabinete. No entanto, os dispositivos comerciais com grande capacidade de armazenamento disponíveis mostram-se não adaptados a aplicações em que seja fundamental a sincronização temporal entre diversos sensores.

Assim, e tendo em conta o referido na secção anterior, é pretendido que o DataLogger a desenvolver apresente as seguintes características:

- Capacidade de armazenamento de dados de um sensor num PenDrive USB genérica – este objectivo é fundamental para que os dados sejam, depois, transmitidos para um computador com elevada eficiência, bastando para tal remover a PenDrive do DataLogger e aplicá-la a uma entrada USB do computador. Este ponto é essencial e inovador pois as soluções adoptadas nos dispositivos comerciais implicar que este processo seja efectuado via um protocolo de transmissão série entre o dispositivo e o computador, em que esse protocolo é muito mais lento que o USB (usado na comunicação com a PenDrive). Mais ainda, e dado que os circuitos integrados de armazenamento de dados (no formato de cartões de memória flash, por exemplo) apresentam uma evolução tecnológica impressionante, a escolha da PenDrive USB apresenta a vantagem acrescida de contribuir para uma maior longevidade do DataLogger pois o protocolo USB de comunicação com PenDrives é genérico e foi feito para estar funcional durante muito tempo;
- Capacidade de comunicação com sensores através de diferentes protocolos série, como RS-232, RS-458, $I^2C$, SPI, entre outros, para ser facilmente adaptável a um vasto leque de sensores;
- Possibilidade de recepção de um sinal de sincronismo temporal tipicamente proveniente de um receptor GPS de muito baixo custo. Este mecanismo é constituído por um sinal de *trigger* (é um impulso de nível TTL gerado no início de cada segundo do tempo GPS, com uma precisão de cerca de 50 ns) e de uma linha de dados RS-232 com a informação temporal de cada disparo de *trigger* (pressupõe a existência de um elemento externo ao DataLogger responsável por gerar o sincronismo, como um receptor GPS, ou de outro DataLogger com a configuração de SyncMaster – ver ponto seguinte);
- Possibilidade de ser o elemento gerador do sinal de sincronismo (adquirindo o papel de SyncMaster) que sincronizará os DataLoggers a este ligados (que terão o papel de SyncSlave);

- O armazenamento dos dados na PenDrive USB deverá ser feito por ficheiros, sendo directamente interpretável como tal quando a PenDrive é inserida no computador de análise dos dados (desta forma torna-se muito simples e prático proceder ao descarregamento dos dados, sendo apenas necessário recolher as PenDrives, inseri-las no computador e copiar/mover os ficheiros;
- Deverá ocupar muito pouco espaço (para tal, o DataLogger deverá ser preferencialmente projectado com componentes SMD);
- Deverá consumir pouca energia;
- Deverá poder ser alimentado a +5V ou, em alternativa, num qualquer valor na gama +7V a +30V.

## 1.3 Estrutura da tese

Este documento está organizado em seis capítulos. No capítulo dois é apresentado o estado da arte referente a dispositivos de armazenamento de dados, nomeadamente os tipos de dispositivos que se encontram actualmente no mercado, seus componentes e características. É também descrito um breve resumo sobre a evolução dos micro-controladores.

No capítulo três é enunciada a selecção do material e ferramentas de maior relevância como o micro-controlador e a aplicação de *software* de desenvolvimento.

No capítulo quatro é descrita a concepção e implementação do DataLogger, nomeadamente a placa do micro-controlador, a placa do DataLogger, o seu firmware e, finalmente, a especificação do conteúdo dos ficheiros armazenados pelo DataLogger.

No capítulo cinco apresentam-se os testes e respectivos resultados efectuados ao dispositivo desenvolvido. Testes como a recepção de um ficheiro de texto enviado pela porta série de um computador, recepção de uma quantidade

aleatória de bytes e a verificação da actualização do relógio do DataLogger através do sinal de GPS de 1PPS (1 pulso por segundo) demonstram o bom funcionamento do protótipo.

Finalmente, no capítulo seis, são apresentadas as conclusões relativas ao trabalho desenvolvido e algumas sugestões para trabalho futuro.

# Capítulo 2

# 2 Estado da Arte

Neste capítulo é primeiramente apresentado o estado da arte sobre os dispositivos que se encontram no mercado para armazenamento de dados de sensores. De seguida, é feito uma breve exposição sobre a evolução dos micro-controladores, tendo esta uma maior incidência em dispositivos da família PIC da Microchip.

## 2.1 Dispositivos de Armazenamento de Dados

Um DataLogger ou DataRecorder é por definição um dispositivo electrónico de registo de dados ao longo do tempo em formato digital. Em aplicações como sismómetros, polígrafos, electrocardiógrafos e outras, os DataLoggers electrónicos vieram substituir os chart recorders. Estes últimos são normalmente dispositivos que recebem dados por meio eléctrico ou mecânico e os registam em papel.

Os dados registados podem ser provenientes de instrumentos ou sensores que, por sua vez, podem ser internos ou externos ao próprio DataLogger. São normalmente baseados em processadores digitais ou até mesmo em computadores. Consoante esta base, podem ser pequenos, alimentados a bateria, portáteis e conter memória e sensores internos ou, mais parecidos com workstations, com o intuito de armazenar grandes volumes de dados, de elevado preço, alimentados a electricidade e imóveis. Os DataLoggers variam desde várias aplicações com um propósito genérico a outros muito específicos que apenas fazem registos de um certo ambiente ou aplicação. Existem DataLoggers

reprogramáveis e outros que permanecem com um limitado número ou mesmo nenhum parâmetro configurável.

Actualmente os DataLoggers estão em constante mutação. Isto porque o modelo original de um dispositivo autónomo está a alterar para um dispositivo que regista informação, mas também tem acesso a comunicação wireless para eventos de alarme, transferência automática dos registos efectuados e para controlo remoto [1].

Da pesquisa elaborada, foram avaliados diversos dispositivos. Muitos dos quais destinados ao registo de dados específicos e com baixa capacidade de armazenamento. Encontra-se DataLoggers portáteis e para aplicações genéricas como o ALL-XM8 Serial DataLogger que se limitam ao registo dos dados recebidos pela porta série sem qualquer tipo de acondicionamento [2] e outros que registam os dados em cartões de memória SD, mas que não inserem etiqueta temporal nos mesmos. Um dispositivo como o Rugged NMEA DataLogger da Brookhouse tem muitas das características pretendidas, no entanto é específico para aplicações de medida de profundidade aquática [3]. Existem também dispositivos produzidos pela MicroDAQ, como o DT80, que possuem características semelhantes às pretendidas mas apresentam um preço excessivamente elevado (2680 dólares) [4]. Alguns têm como base ou são dependentes de computadores, logo demasiado grandes e com portabilidade reduzida para aplicações de campo como é o caso do Advanced NMEA DataLogger [5]. O Universal SD DataLogger, embora seja limitado na gama de sensores, possui recursos aceitáveis, mas de modo a ser utilizado autonomamente, tem de ser inicializado por um computador, o que não o torna síncrono com outros DataLoggers semelhantes. [6].

Como enunciado no capítulo 1, o DataLogger desenvolvido, terá utilidade imediata no projecto de investigação financiado pela FCT denominado INSHORE. Assim, características como portabilidade, dimensão reduzida, baixo consumo de energia, capacidade de armazenamento elevada, introdução de etiqueta temporal nos registos e possibilidade de comunicar com diversos sensores através de

protocolos série são essenciais, mas não são satisfeitos pelos dispositivos anteriormente citados.

## 2.2 Evolução dos Micro-controladores

Nos dias de hoje é raro o sistema ou aplicação que não tenha embutido um micro-controlador. Os micro-controladores têm grandes vantagens, tais como, baixo custo, baixo consumo, portabilidade, reconfiguração por software, entre outras. Actualmente, estes encontram-se embutidos dentro de inúmeros sistemas ou dispositivos, por exemplo, se um forno de microondas possui um visor LCD (*Liquid Crystal Display*) e um teclado é porque contém pelo menos um micro-controlador a controlar estes (e outros) dispositivos. Basicamente, qualquer produto ou dispositivo que interaja com o utilizador possui um micro-controlador interno.

Uma vez que o micro-controlador é uma peça fundamental na concepção do DataLogger, foi necessário fazer um apanhado dos micro-controladores disponíveis no momento no mercado para se seleccionar o mais adequado. Assim, apresenta-se nas secções seguintes esse estudo, acompanhado pela recente evolução das tecnologias mais relevantes neste domínio, assim como se apresenta a arquitectura de um micro-controlador genérico (que salienta as suas características mais importantes, determinantes na comparação e selecção de micro-controladores).

### 2.2.1 Componentes de um micro-controlador

O micro-controlador, também conhecido como MCU ou µC, combina os recursos fundamentais disponíveis de um micro-computador, tais como o CPU (*Central Processing Unit*), memória e recursos de I/O (*Input/Output*) num único

chip, sendo estes componentes interligados por grupos de linhas eléctricas, denominadas de barramento. A figura 1Figura 1Figura 1 apresenta o diagrama de blocos de um micro-controlador genérico.

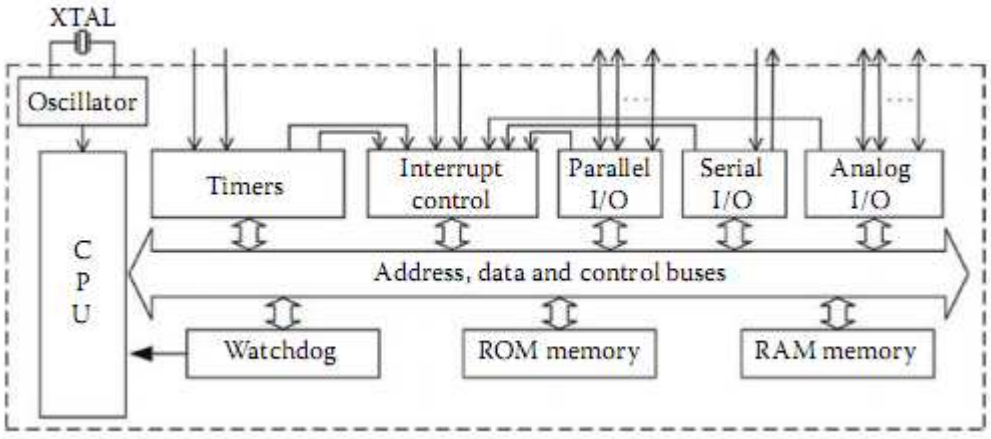


**Figura 1: Diagrama de blocos de um micro-controlador genérico [7].**

Os micro-controladores têm um oscilador para gerar o sinal necessário para sincronizar todas as operações internas. O cristal de quartzo (XTAL) é normalmente usado devido à sua estabilidade em alta frequência. A frequência do oscilador está directamente ligada à velocidade a que os programas são executados.

Semelhante aos micro-computadores, o CPU é o cérebro do micro-controlador. Utiliza as instruções do programa a partir da sua localização em memória, uma por uma, interpreta-as ou descodifica-as e executa-as. O CPU também inclui os circuitos ALU (*Arithmetic Logic Unit*), circuito digital que realiza operações aritméticas e lógicas.

Quanto à memória do micro-controlador, esta armazena tanto o programa de instruções, como os dados. Qualquer micro-controlador possui dois tipos de memória: *Random-Access Memory* (RAM) e *Read-Only Memory* (ROM). A memória RAM pode ser de leitura e de escrita. É uma memória volátil, o que significa que os seus dados são perdidos quando a alimentação é desligada. Por outro lado, a memória ROM é uma memória apenas de leitura e não volátil. Existem vários

tipos de tecnologias usadas para a memória ROM, tais como: EPROM (*Erasable Programmable Read-Only Memory*), EEPROM (*Electrical Erasable Programmable Read-Only memory*), OTP (*One-Time Programmable*) e *flash*.

Os recursos I/O são o meio de comunicação com o mundo exterior, assim são muito importantes nos micro-controladores. Consistem em portas séries, portas paralelas, timers e interrupções. Alguns micro-controladores também possuem entradas analógicas e saídas associadas a ADC (*Analog to Digital Converter*) e DAC (*Digital to Analog Converter*). Os dispositivos de I/O podem ir de um simples pino digital do componente a uma interface USB (Universal Serial Bus) ou Ethernet nos mais avançados.

O WatchDog Timer (WDT) é um recurso que pode ser encontrado na maioria dos micro-controladores. É constituído por um oscilador e por um contador binário de N bits. Quando o contador alcança o seu valor máximo, a saída do contador é activada e é dado um sinal de reset ao micro-controlador. Quando o programa é executado correctamente, o contador do WDT nunca atinge o máximo. No entanto, se o programa bloquear, parando a sua a execução, o contador WDT irá alcançar o seu valor máximo, enviando um sinal de *reset* ao micro-controlador, que vai fazer com que o programa execute do início novamente [7].

### 2.2.2 Fabricantes de micro-controladores

Actualmente existem vários fabricantes de micro-controladores, sendo os mais importantes a Microchip, Texas Instruments, Intel e ATMEL.

Mesmos dispositivos da mesma família possuem capacidades de I/O e memórias de tamanho diferentes. Por exemplo, todos os micro-controladores da família 8051 (MCS51) têm um CPU similar e executam as mesmas instruções. No entanto, vários membros da mesma família têm diferentes tipos e números de portas de I/O e também diferentes tipos e tamanhos de memória [7].

A Microchip Technology Inc. é um grande fabricante de micro-controladores que produz os bem conhecidos micro-controladores PIC. Estes são uns dos mais populares, sendo usados em muitas aplicações comerciais e industriais. São vendidos mais de 120 milhões de dispositivos por ano [8].

### 2.2.3 Características dos micro-controladores PIC

Todos os micro-controladores PIC são baseados na arquitectura *Harvard*. Esta é caracterizada por ter diferentes memórias de programa e de dados. Como é comum na maioria dos micro-controladores, o tamanho da memória do programa é maior que o tamanho da memória de dados. A memória do programa está organizada em palavras de 12, 14, 16 e 32 bits, enquanto que a memória de dados é baseada em registos de 8 bits. O acesso a dispositivos I/O é realizado através de alguns registos na memória de dados chamados *Special Function Registers* (SFRs). Vários micro-controladores PIC também têm memórias adicionais, EEPROM, para armazenar dados (alteráveis, tipicamente referentes a parâmetros de configuração) em modo não volátil.

Todos os micro-controladores PIC são micro-controladores RISC (*Reduced Instruction Set Computer*), proporcionando uma grande rapidez e flexibilidade, com fácil migração de apenas 6 para até 100 pinos em diversos encapsulamentos (SOT23, DIP, SOIC, etc.), e de 384 bytes para 512 kbytes de memória de programa.

Todos os micro-controladores PIC usam *pipelining* para executar instruções. Este *pipelining* consiste em executar as instruções do programa em etapas paralelas, tornando possível a execução de uma instrução por *machine cycle* (cada *machine cycle* contém quatro ciclos do oscilador principal).

Outra característica especial dos micro-controladores PIC é a implementação da pilha (*stack*). Aqui a pilha não faz parte da memória de dados, mas tem o seu próprio espaço independente e, portanto, um tamanho finito. O tamanho da pilha depende de modelo para modelo PIC.

Os micro-controladores PIC têm uma grande variedade de dispositivos I/O. Possuem porta paralela, timers, portas série síncronas e assíncronas, conversores ADC e DAC, e saídas PWM (*Pulse Width Modulators*).

Todos os micro-controladores PIC têm um contador de segurança, que é conhecido como *watchdog timer*. Este contador pode ser configurado com bits específicos quando o micro-controlador é programado. Outras configurações de bits podem ser usadas para proteger a memória do programa contra cópia não autorizada. [7]

### 2.2.4 Micro-controladores PIC32

A Microchip, em Novembro de 2007, apresentou a nova família PIC32MX como micro-controladores de 32-bits. Os primeiros modelos em produção PIC32MX3xx e PIC32MX4xx são compatíveis e partilham o mesmo conjunto de periféricos com os dispositivos da família PIC24FxxGA0xx de 16-bit, o que permite o uso das mesmas livrarias, software e ferramentas de hardware.

A arquitectura do PIC32 traz um conjunto de novos recursos sendo os mais importantes [9]:

- Velocidade de execução com 80 MIPS;
- Memória flash até 512 Kbytes;
- Uma instrução por ciclo de *clock*;
- Processador com *cache*;
- Permite execução a partir da memória RAM;
- Capacidade USB de Full Speed Host/*Dual Role* e OTG;

Pode-se verificar pela figura 2, que apresenta o diagrama de blocos da família PIC32MX4xx, diversas características deste micro-controlador.

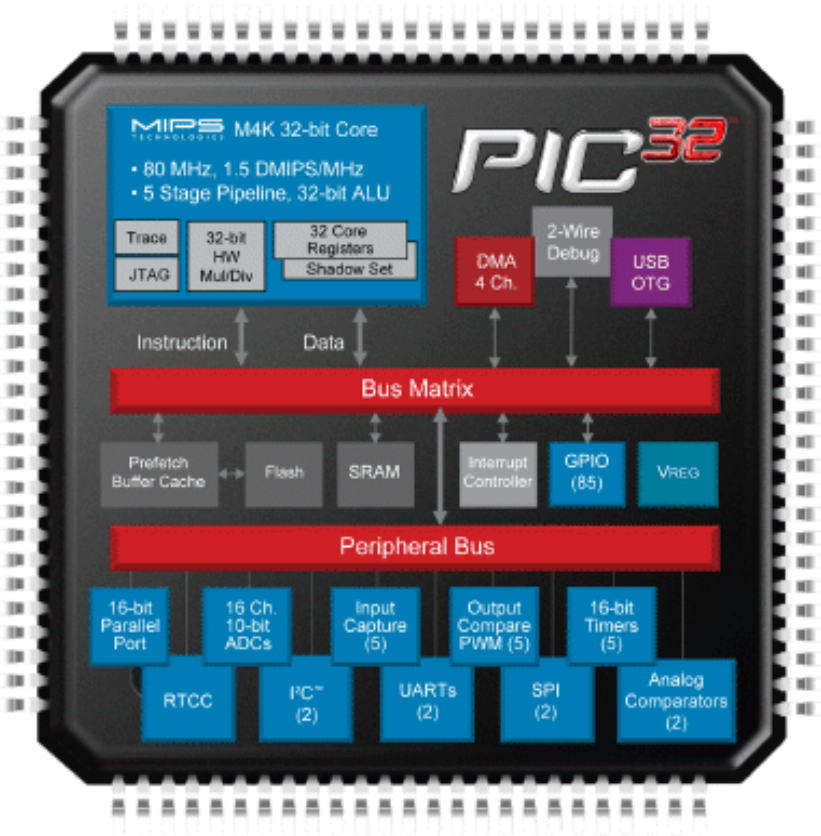


**Figura 2: Diagrama de blocos do PIC32 [10].**

# Capítulo 3

# 3 Escolha de material e ferramentas

Após um sumário estudo e pesquisa relativamente ao protocolo USB, deu-se início ao desenvolvimento do projecto. Assim, nesta fase, a escolha de ferramentas e material a usar é o primeiro passo que pode ser dividido em três partes que passo a citar:

- Secção do micro-controlador a utilizar;
- Escolha da aplicação de *software* e dispositivo de interface de programação do micro-controlador;
- Escolha da aplicação de *software* para desenvolvimento da placa PCB do micro-controlador.

Refere-se também que o trabalho realizado nesta fase foi feito em conjunto com o aluno Nuno Miguel Castro Gonçalves, no âmbito dos seus trabalhos de dissertação de mestrado denominado "Desenvolvimento de uma Interface de Sinais Analógicos".

## 3.1 Escolha do micro-controlador

Para o desenvolvimento de sistemas electrónicos baseados em micro-controladores, a escolha correcta de um dispositivo que se enquadre com os objectivos pretendidos, pode fazer toda a diferença no sucesso do projecto.

Optar-se por um micro-controlador da família da Microship foi uma decisão fácil, pois actualmente situam-se entre os que são mais usados a nível profissional e são também alvo de estudo em disciplinas anteriores. As características principais que levaram a esta selecção foram a flexibilidade que é reconhecida nestes dispositivos, a existência de ferramentas de programação e *debug* gratuitamente disponíveis na internet, e a experiência prévia na programação e utilização destes dispositivos obtida em disciplinas do curso de Mestrado Integrado em Engenharia Electrónica e Telecomunicações.

Dado que o PIC32 foi lançado no ano de 2007, fazendo dele a mais recente família de processadores da Microchip, este irá conferir ao DataLogger uma longevidade considerável usando a mais moderna tecnologia dos micro-controladores e contado com funcionalidades como *system clock* do micro-processador de elevada frequência (até 80 MHz), interface USB 2.0 (*host* e *device*), UART, entre outras.

Sendo um dos principais objectivos para ambas as dissertações, a implementação do protocolo USB quer do ponto de vista do *host* no caso do presente trabalho, quer do ponto de vista do *device* para a dissertação do autor Nuno Gonçalves, a selecção do micro-controlador da família PIC32 recaiu apenas em três modelos.

- PIC32MX42
- PIC32MX44
- PIC32MX46

Devido a factores como servir os objectivos de ambos os trabalhos, possuir uma boa relação preço/qualidade, ter um menor número de pinos para facilitar a criação do PCB e ter o máximo de memória, a escolha final incidiu sobre o PIC32MX440F512H (cujo *pinout* se apresenta na figura 3) que tem os seguintes recursos:

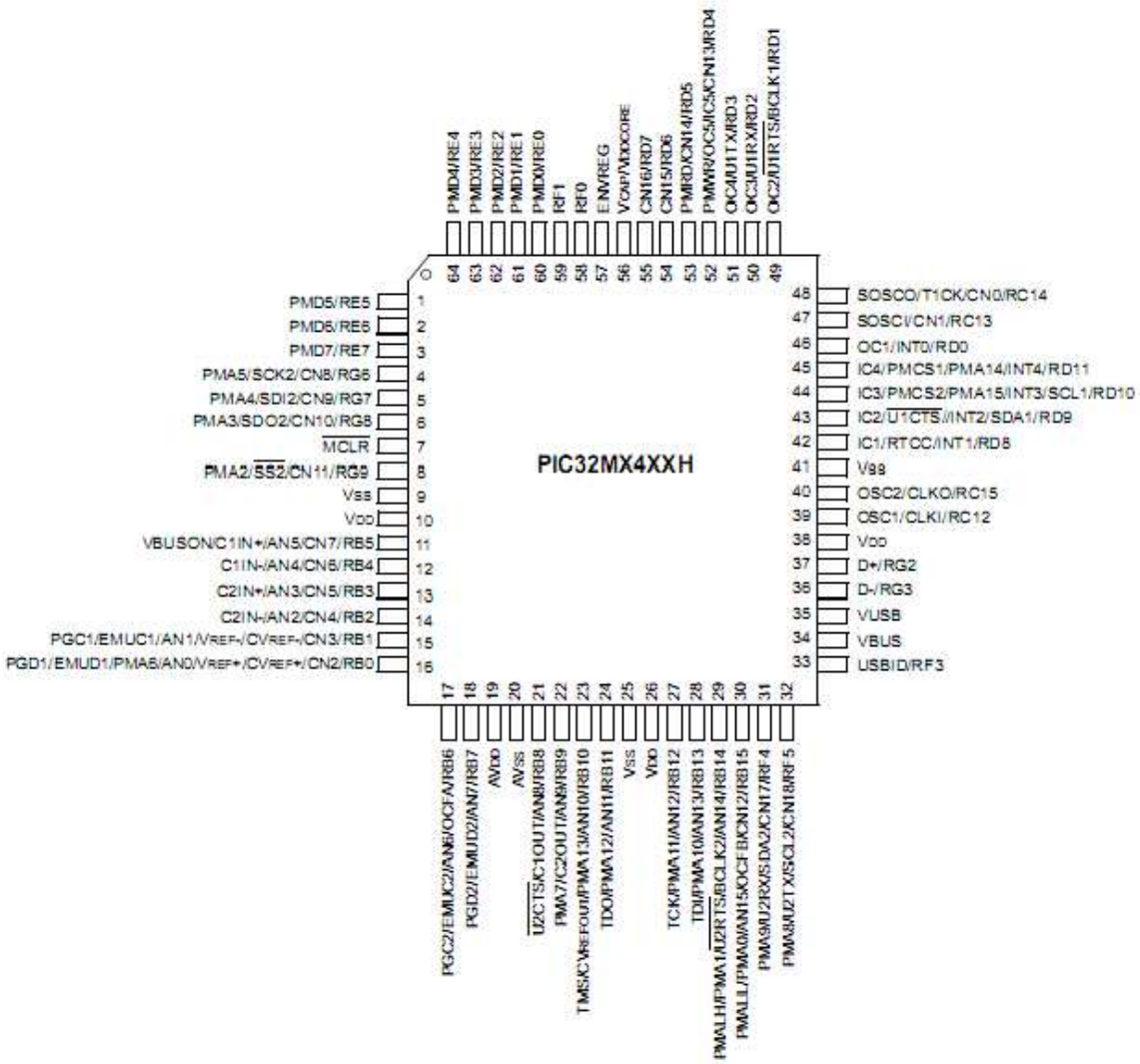


**Figura 3: *Pinout* de chips da família PIC32MX4xxH [11].**

- Micro-controlador com 64 pinos;
- *System Clock* com frequência máxima de 80MHz;
- Osciladores internos de 8MHz e 32KHz;

- *Hardware* RTCC (*Real Time Clock and Calendar with Alarms*);
- Memória: 512 kbytes de memória de programa (*Flash*), 32 kbytes de memória de dados (RAM);
- USB *On-The-Go;*
- 5 Interrupções Externas;
- 5 *Timers* digitais de 16 *bits*;
- 5 Módulos *Input Capture*
- 5 Módulos *Output Compare*/Std. PWM
- 4 Canais DMA (*Dynamic Memory Access*)
- 51 pinos *Input/Output*
- Porto Paralelo PMP
- 16 Canais Analógico/Digital
- 1 Módulo SPI
- 2 Módulos compatíveis com $I^2C$
- 2 Módulos UART (*Universal Asynchronous Receiver Trasmitter*);
- 1 Módulo ADC de 16 canais, 10 bits;

Para mais informações sobre as funcionalidades do micro-controlador PIC32MX440F512H ou detalhes de alguma característica mais relevante, aconselha-se a consulta de [11].

## 3.2 Ferramenta de programação do micro-controlador

Hoje em dia existem várias formas de programar e reprogramar um micro-controlador. Para o desenvolvimento de ambos os trabalhos, decidiu-se escolher um método de programação *in-circuit* que suportasse também o modo *debug.* Assim sendo, podia-se implementar um circuito de programação/*debug* na placa do micro-controlador ou adquirir um dispositivo com essa funcionalidade fornecido pela Microchip. Optou-se pelo último, dado que é método mais simples

e eficiente. O funcionamento destes programadores é relativamente simples, dado que apenas é necessário ligar o micro-controlador ao dispositivo de programação através de um reduzido conjunto de pinos.

Da pesquisa efectuada, foram seleccionados três dispositivos de programação/*debug* cujas características principais passo a citar:

| MPLAB ICD3 | PICkit 2 | PICkit 3 |
|---|---|---|
|  |  |  |
| Ligação USB High Speed | Ligação USB Full Speed | Ligação USB Full Speed |
| Suporta todos os tipos de emulação | Ligação com aplicação através do mecanismo ICSP | Ligação com aplicação através do mecanismo ICSP |
| Execução e *debug* em tempo real | Execução em tempo real | Execução em tempo real |
| Preço elevado | Preço reduzido | Preço reduzido |

**Tabela 1: Ferramentas de Programação/debug**

Todos os dispositivos satisfazem os requisitos pré-definidos, assim começou-se por descartar o ICD3 devido ao seu elevado preço comparativamente aos restantes. Dado que a própria Microchip aconselha o uso do PICkit3 para a programação de micro-controladores da família PIC32 e como as diferenças existentes entre este e o PICkit2 não são significativas, optou-se pelo PICkit3.

A Microchip também faculta o MPLAB® como *software* de desenvolvimento (programação e *debug* de código para micro-controladores da Microchip). Este IDE (*Integrated Development Environment*) integra um conjunto de ferramentas livres para o desenvolvimento de aplicações embutidas que utilizem micro-controladores PIC e dsPIC da Microchip. MPLAB IDE corre como um aplicativo de 32 bits em MS Windows®, é fácil de usar e inclui uma série de componentes de

software livre para um rápido desenvolvimento de aplicações e *debug*. Serve, também, como uma interface gráfica única para o utilizador, não sendo necessário recorrer a *software* adicional ou a ferramentas de terceiros [12]. De salientar que este IDE oferece uma interface de programação em linguagem C, o que facilita imenso a programação dos micro-controladores.

## 3.3 Ferramenta de desenvolvimento de PCB

São diversas as aplicações existentes para o desenvolvimento e desenho de placas de circuito impresso. Devido à sua ampla utilização a nível profissional, foi sugerido o *software* Cadence OrCAD®. Esta é uma aplicação multifacetada, possuindo uma ampla lista de ferramentas. Elaborar uma placa de circuito impresso implica a criação do esquemático do circuito para posteriormente desenhar o *layout* da placa. Assim sendo, foram utilizados os seguintes recursos deste programa: OrCAD Capture e OrCAD Layout. Destinando-se o primeiro à criação do esquemático do circuito e o segundo ao desenho da placa [13].

Finalmente, este software foi escolhido, pois ter a competência de trabalhar com ele é uma mais-valia para um engenheiro que trabalhe na área de desenvolvimento de *hardware*.

# Capítulo 4

# 4 Concepção e Implementação do DataLogger

Neste capítulo são descritos todos os passos dados desde a concepção à implementação do dispositivo DataLogger.

Inicialmente é apresentada a placa do micro-controlador e respectivas componentes, segue-se a placa do DataLogger e o firmware desenvolvido. Finalmente, é especificado o conteúdo dos ficheiros armazenados pelo DataLogger.

## 4.1 A Placa do Micro-controlador

Devido às características físicas do micro-controlador PIC32MX440F512H, a criação de uma placa independente de interface deste dispositivo com outros circuitos electrónicos iria agilizar em grande escala os testes do protótipo. O objectivo desta placa de interface com o micro-controlador (denominada de PIC32 *kit board*) foi o de criar uma placa com interface DIP contendo os componentes básicos para o funcionamento do PIC32 (tais como cristais para o *clock* externo, interface de programação, circuito de *reset*, e filtragem das linhas de alimentação) para que esta placa possa ser usada em vários projectos que envolvam a criação de dispositivos controlados por este micro-controlador. Deste modo, o desenvolvimento desta placa fez com que o protótipo do DataLogger pudesse ser

implementado sem recorrer a PCBs com componentes SMD, podendo ser rapidamente implementado com componentes de furação do tipo DIP, cujo manuseamento é substancialmente mais simples. Assim sendo, obteve-se um célere teste de funcionalidades e uma rápida prototipagem do DataLogger.

A referir também que o desenvolvimento desta placa de interface com o micro-controlador foi um trabalho conjunto com o aluno Nuno Miguel Castro Gonçalves, no âmbito dos trabalhos de dissertação de mestrado denominado "Desenvolvimento de uma Interface de Sinais Analógicos" ", dado que esta placa de interface do PIC32 foi utilizada, também, no respectivo trabalho.

Para a criação do desenho da placa de circuito impresso do micro-controlador, a ferramenta utilizada foi Cadence OrCAD®. Esta é constituída por diversas aplicações, das quais apenas são necessárias duas para o desenvolvimento de PCBs. Estas são o OrCAD Capture®, que se limita ao desenho do esquemático do circuito e o OrCAD Layout®, responsável pelo desenho da placa em si e pela criação dos *Gerber Files*.

O desenvolvimento deste género de placas é basicamente constituído por quatro passos. Começa-se pela criação do esquemático do circuito, de seguida desenha-se o *layout* da placa e gera-se os respectivos *Gerber Files*. O quarto passo é fazer a impressão da placa e, finalmente, soldam-se os componentes na mesma.

Com a aplicação OrCAD Capture® desenhou-se o esquemático para que seja possível, de um modo simples, criar as ligações entre os vários componentes. Antes de avançar para o *layout*, associou-se um *footprint* a cada componente. A figura 4 demonstra o esquemático criado.

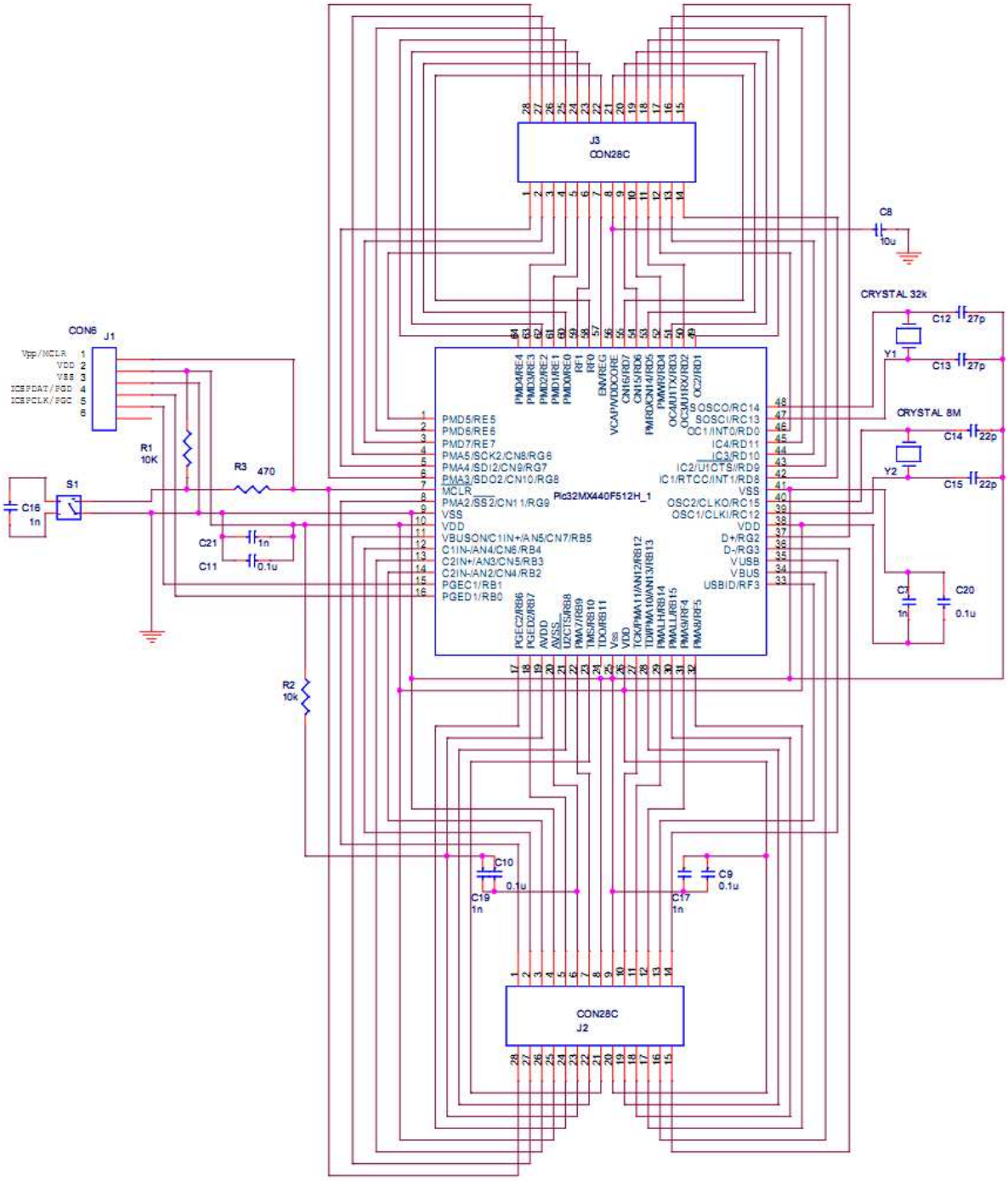


**Figura 4: Esquemático da placa do micro-controlador.**

De seguida, desenhou-se o *layout* da placa com a aplicação OrCAD Layout ®. Nesta fase foi necessário definir a localização física dos componentes e das pistas. Esta placa é constituída por duas camadas de cobre, a superior e a inferior, representadas respectivamente a azul e a vermelho na figura 5. Excepto os *headers*

de ligação a outras placas, todos os componentes estão colocados na camada superior, servindo a camada inferior apenas para a passagem das pistas. Na figura 5 é possível ver a constituição do *layout* da PCB do PIC32 *kit board*.

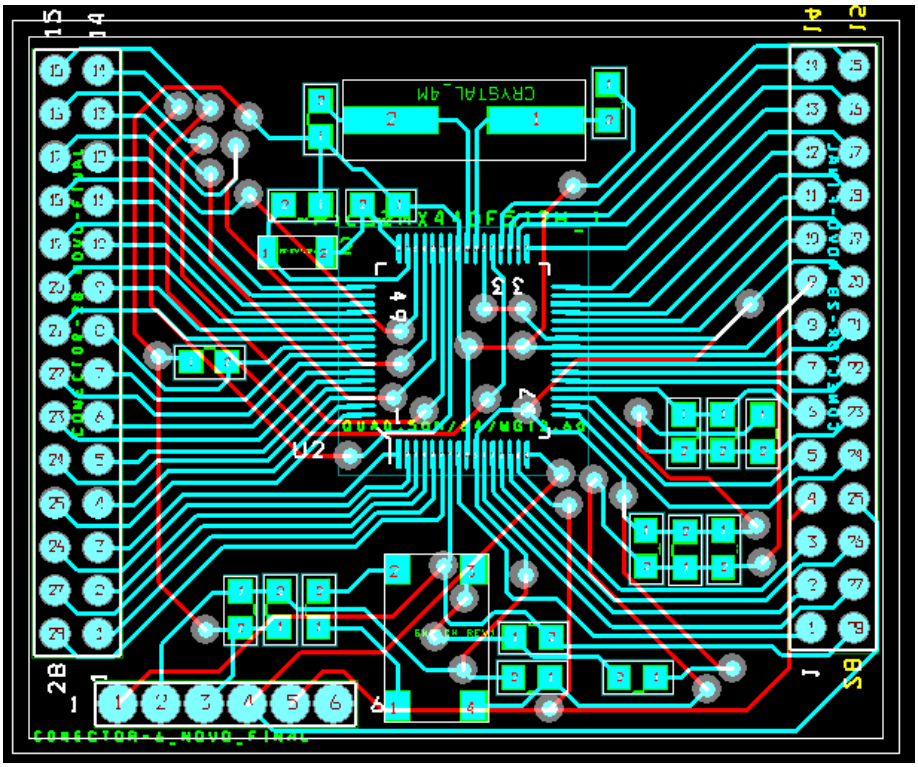

**Figura 5: *Layout* da placa do PIC32 *kit board*.**

Seguidamente procedeu-se à impressão do *layout* em PCB, cujo resultado pode verificar-se na figura 6.

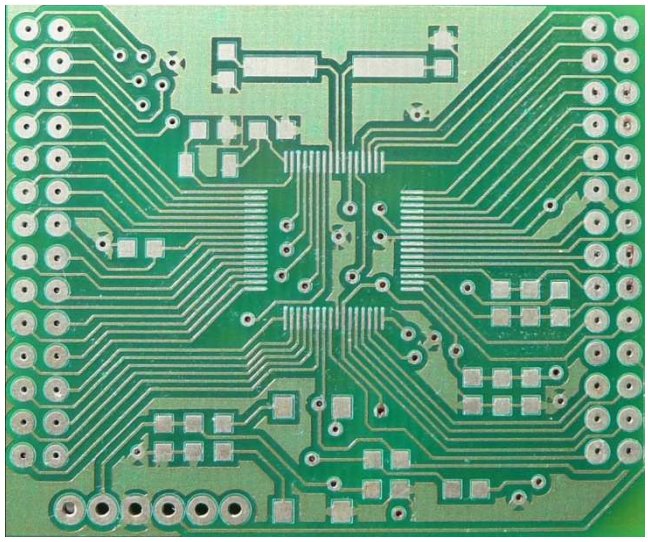

**Figura 6: PCB do PIC32 *kit board*.**

A placa final com todos os componentes apresenta-se na figura 7.

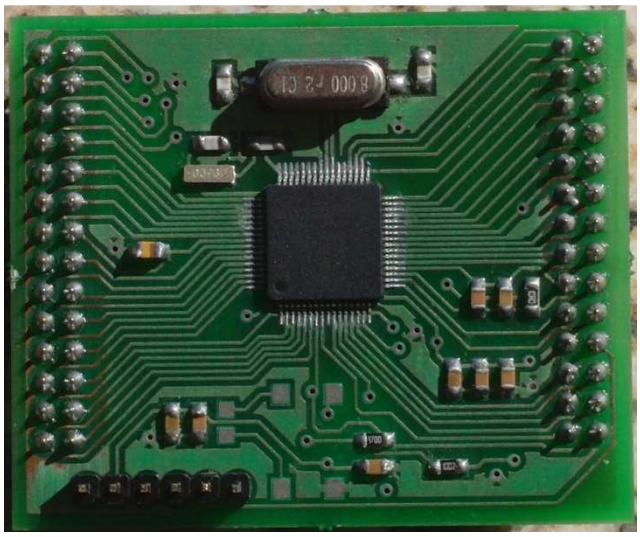

**Figura 7: Vista do PIC32 *kit board*, no seu estado final.**

Tratando-se o PIC32MX440F512H de um micro-controlador recente e com muitas potencialidades, também é propósito desta placa que auxilie e possa ser utilizada futuramente em outras diversas aplicações, nomeadamente trabalhos de dissertação de mestrado de outros colegas.

## 4.2 A placa do DataLogger

Neste subcapítulo apresentam-se as várias fases da concepção de placa do DataLogger. De forma a facilitar a experimentação e o aperfeiçoamento desta, a implementação inicial foi efectuada em *breadboard* e, no final, foi criado um protótipo em placa pré-perfurada.

De seguida, na figura 8, apresenta-se o diagrama de blocos que descreve a arquitectura concebida para o DataLogger, expondo o princípio de funcionamento do DataLogger.

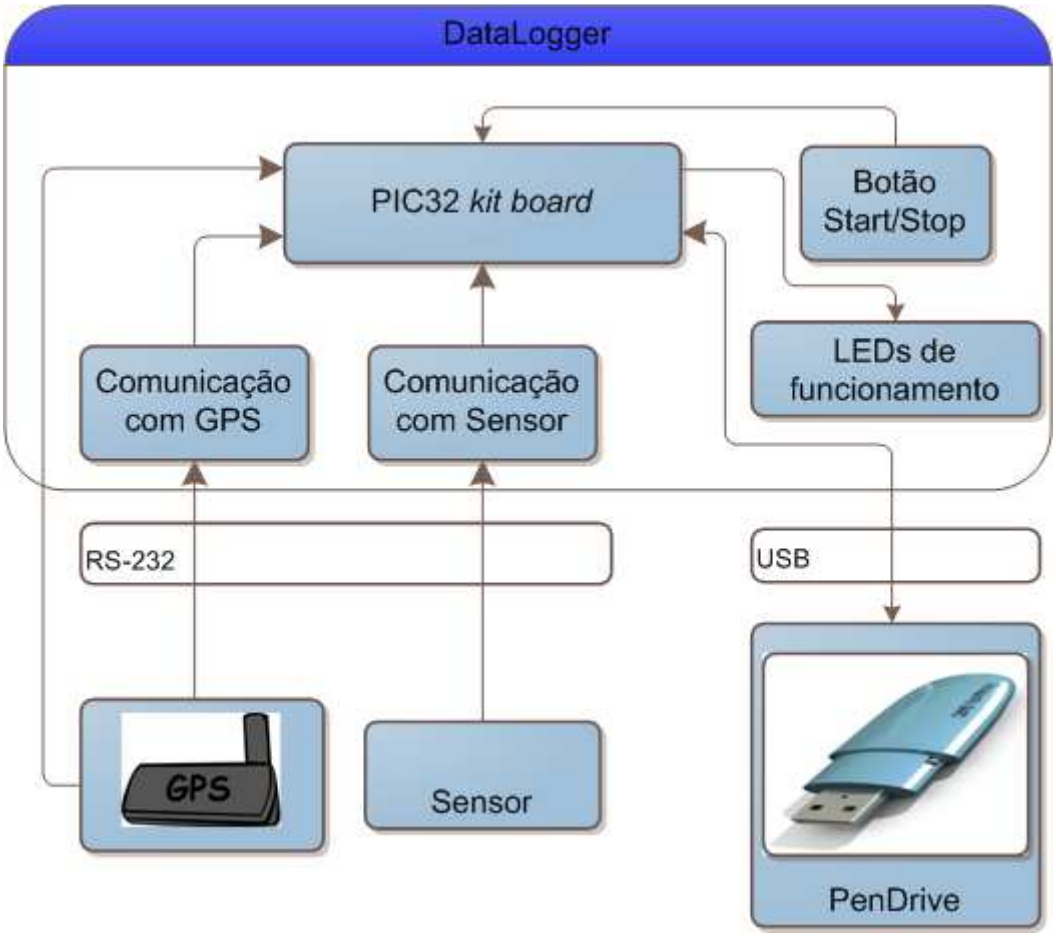


**Figura 8: Diagrama de blocos da placa do DataLogger.**

Como se pode ver pela figura 8, a placa do DataLogger é constituída por um módulo de comunicação com o sensor, um segundo módulo de comunicação com um receptor GPS que fornece a base de tempo ao DataLogger (tempo GPS) segundo a qual este sincroniza a recepção dos dados (colocando uma etiqueta temporal em cada conjunto de dados recebidos), um módulo de visualização do estado de operação do DataLogger (através de um conjunto de LEDs) e, por fim, o módulo de controlo da gravação (que é efectuado por um botão que, quando pressionado a primeira vez dá comando de início à gravação de dados, e quando novamente pressionado indica que essa gravação deve ser interrompida).

O circuito implementado consiste em:

- Circuito de alimentação;
- Circuito de comunicação com o receptor GPS;
- Circuito de comunicação com o computador;
- Circuito de ligação à PenDrive;
- Circuitos de LEDs e do botão;

De seguida é descrita a composição de cada um destes circuitos, enunciando os componentes utilizados.

### 4.2.1 Circuito de alimentação

O circuito de alimentação é essencialmente constituído por um conversor DC–DC e por uma ficha de alimentação.

O conversor DC-DC escolhido foi o TracoPower® tsr1 – 2433 que possui as seguintes características:

- Modo de operação do tipo *Step Down Switching Regulator*;
- Tensão de entrada mínima de 5.5V e máxima de 32V;

- Tensão de saída de 3.3V;
- Corrente máxima de saída de 1A;
- Encapsulamento do tipo SIP;
- Protecção contra curto-circuito;
- Condensadores de filtragem incorporados.


**Figura 9: TracoPower® tsr1 – 2433.**

Na figura 10 é possível ver o esquemático da alimentação e os respectivos componentes.


**Figura 10: Esquemático do circuito de alimentação.**

## 4.2.2 Circuito de comunicação com o receptor GPS

Este circuito tem como intuito implementar uma comunicação série com protocolo RS-232.

Como se pode ver na figura 11, o circuito de comunicação com o receptor GPS, tem um *transceiver* RS-232 e uma ficha do tipo DB9.

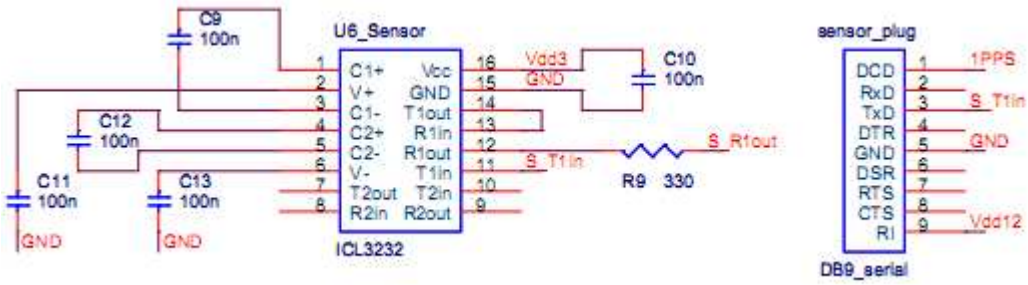


**Figura 11: Esquemático do circuito de comunicação com o receptor GPS.**

O *transceiver* utilizado foi o ICL3232CPZ que é produzido pela Intersil®. Este possui dois canais de comunicação, o encapsulamento é do tipo DIP16 e necessita de ser alimentado com uma tensão no intervalo de 3V a 5.5V. Este *transceiver* contém ainda um conversor de tensão do tipo *charge-pump* que apenas carece de cinco condensadores externos de 100nF.

**Figura 12: *Transceiver* ICL3232CPZ.**

## 4.2.3 Circuito de comunicação com o sensor

Este circuito permite executar a comunicação com um sensor no protocolo RS-232.

Dado que o protocolo de comunicação é igual ao utilizado pelo receptor GPS, os componentes utilizados também são iguais. No entanto, como se pode ver pela figura 13, o esquemático é ligeiramente diferente. Isto deve-se ao facto de terem sido implementadas outras ligações (principalmente controlo da comunicação por *hardware*) para o caso de serem necessárias.

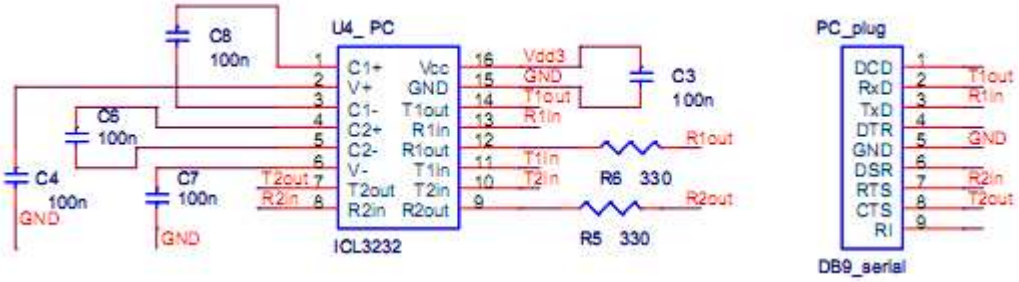


**Figura 13: Esquemático do circuito de comunicação com o sensor.**

### 4.2.4 Circuito de ligação à PenDrive

Uma das principais características do micro-controlador escolhido é possuir um módulo USB incorporado. Assim, como se pode ver pela figura 14, o circuito de ligação à PenDrive é bastante simples. Este consiste apenas num conversor DC-DC e numa ficha USB do tipo A.

Para implementar a configuração de USB *Embedded Host* é necessário aplicar, respectivamente, 5V e 3.3V aos pinos VBUS e VUSB do PIC32. Deve-se ainda aplicar um condensador de 100nF entre cada um destes pinos e a massa.


**Figura 14: Esquemático de ligação à PenDrive e respectiva alimentação.**

Os *devices* USB exigem uma alimentação de 5V, assim sendo, o conversor DC-DC escolhido foi o TracoPower® tsr1 – 2450. Este conversor é da mesma família do escolhido para o circuito de alimentação, apenas divergindo nas seguintes características:

- Tensão de entrada mínima de 6.5V e máxima de 32V;
- Tensão de saída de 5.0V.


**Figura 15: TracoPower® tsr1 – 2450.**

### 4.2.5 Circuitos de LEDs e do botão;

Embora os circuitos de LEDs e o do botão sejam dois circuitos distintos, estes podem ser considerados como circuitos de funcionamento. Isto porque um é o módulo de visualização do estado de operação e, o outro, é o módulo de controlo da gravação.

Como se pode ver pela figura 16, os LEDs estão ligados, através de uma resistência de 470Ω, aos pinos RF1 e RF0 do PIC32.

A configuração do botão permite provocar uma interrupção na linha INT3 do PIC32. Esta interrupção é que vai indicar o início ou o fim da gravação.

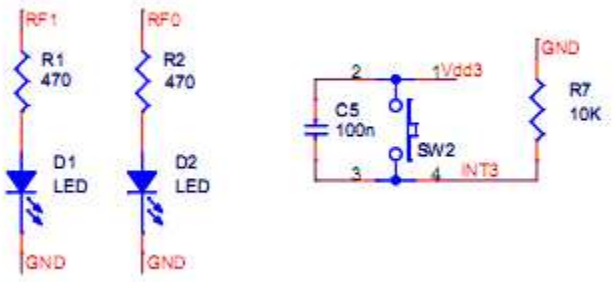


**Figura 16: Esquemático dos circuitos de LEDs e do botão.**

### 4.2.6 Circuito global

Conjugando os circuitos anteriores, obtém-se o esquemático global do DataLogger presente na figura 17. Foram ainda adicionados dois *headers* (de 2x14 pinos) para fazer a ligação com o PIC32 *kit board*.

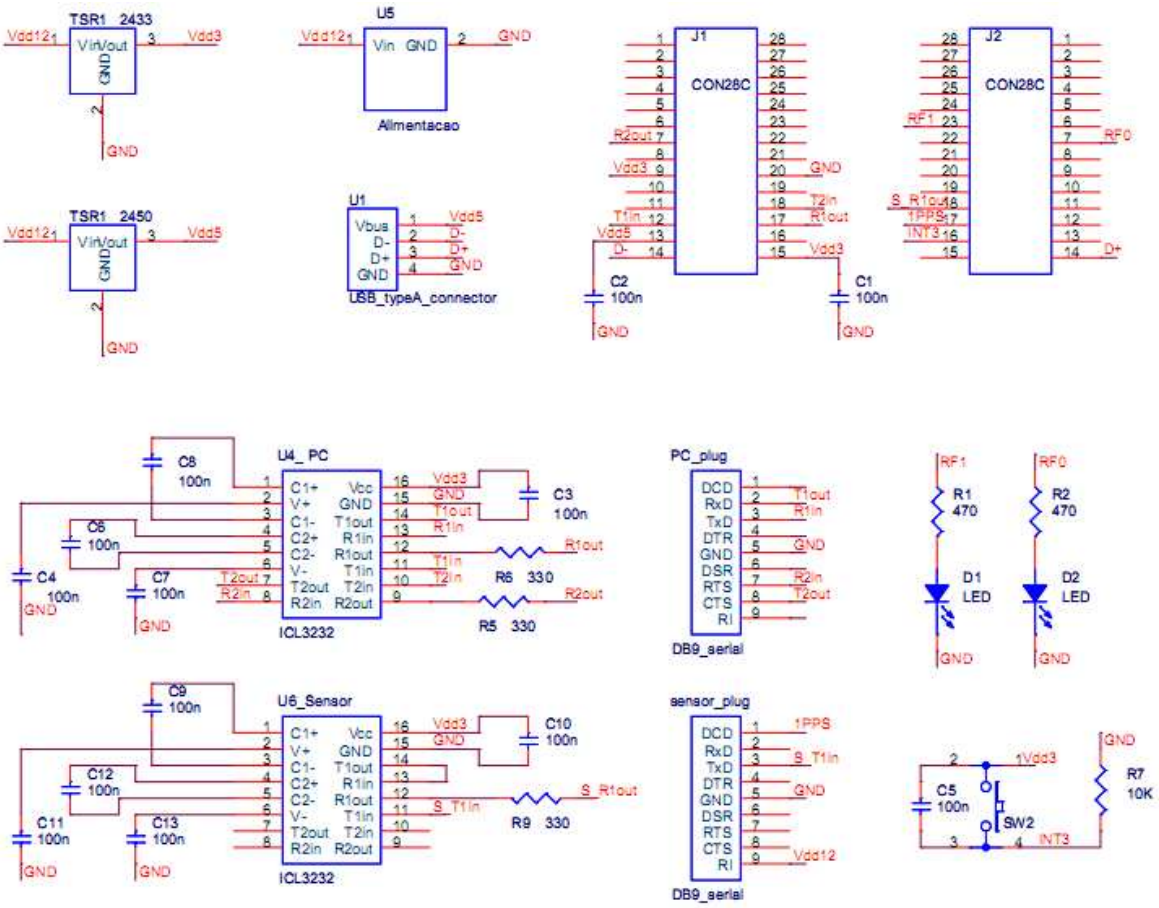


**Figura 17: Esquemático da placa do DataLogger**

## 4.2.7 Evolução da placa do DataLogger

A evolução da placa do DataLogger foi composta por duas fases principais, a fase de desenvolvimento e a fase de protótipo.

Inicialmente, e para facilitar o aperfeiçoamento, foi criado um DataLogger em *breadboard*. Desta forma, tornou-se simples a experimentação de diversas configurações e alguns melhoramentos do DataLogger. A indicar também que, o PIC32 *kit board*, não foi desenhado para ser directamente compatível com *breadboards.* Desta forma, foi necessário criar dois adaptadores, presentes na figura 18, para que esta implementação fosse possível.

**Figura 18: Adaptadores do PIC32 *kit board* para *breadboard*.**

Pela figura 19 é possível ver o estado final desta implementação do DataLogger.

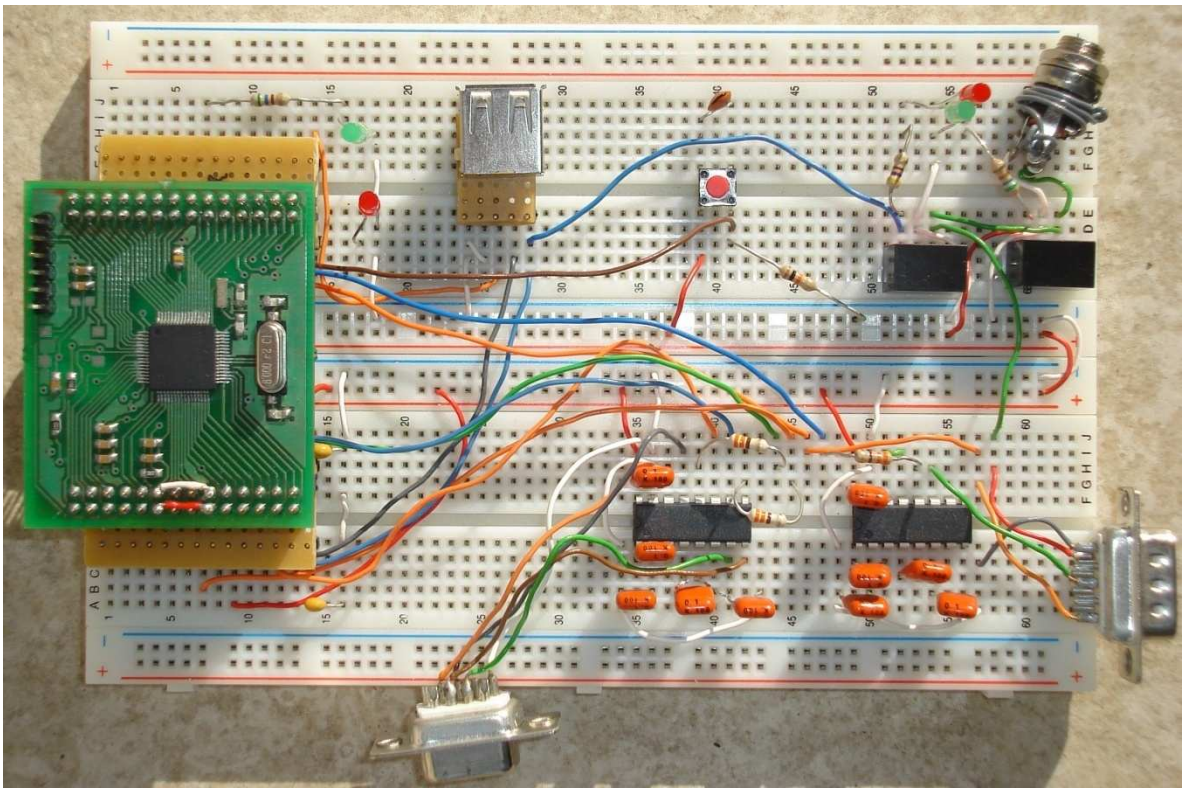

**Figura 19: DataLogger em *breadboard*.**

Para se obter um modelo final, foi criado um protótipo. Esta fase consistiu em implementar o DataLogger em placa pré-perfurada e, como se pode ver pela figura 20, todos os componentes e respectivas ligações foram soldados.

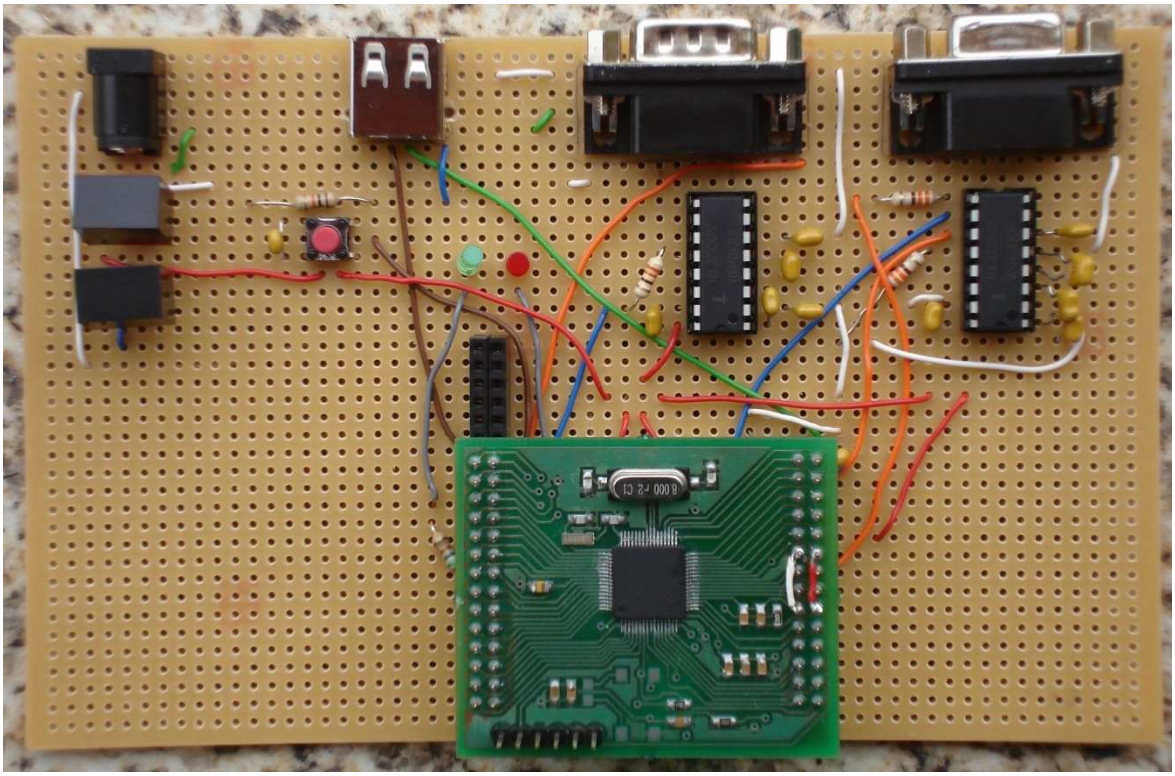

**Figura 20: DataLogger em placa pré-perfurada.**

Refere-se ainda que os elementos sensíveis (PIC32 *kit board*, os conversores DC-DC e os *transceivers* ICL3232CPZ) não estão soldados directamente à placa pré-perfurada, mas sim ligados através dos correspondentes *headers*, proporcionando, assim, uma fácil troca dos mesmos.

## 4.3 *Firmware* do DataLogger

Neste subcapítulo apresenta-se uma vista geral do código desenvolvido para o DataLogger. Posteriormente é feita uma descrição pormenorizada de cada bloco de código, enunciando os seus intuitos e funcionalidades. A figura 21 apresenta um diagrama de blocos da arquitectura geral do código desenvolvido (denominado de *firmware*) para o micro-controlador do DataLogger.

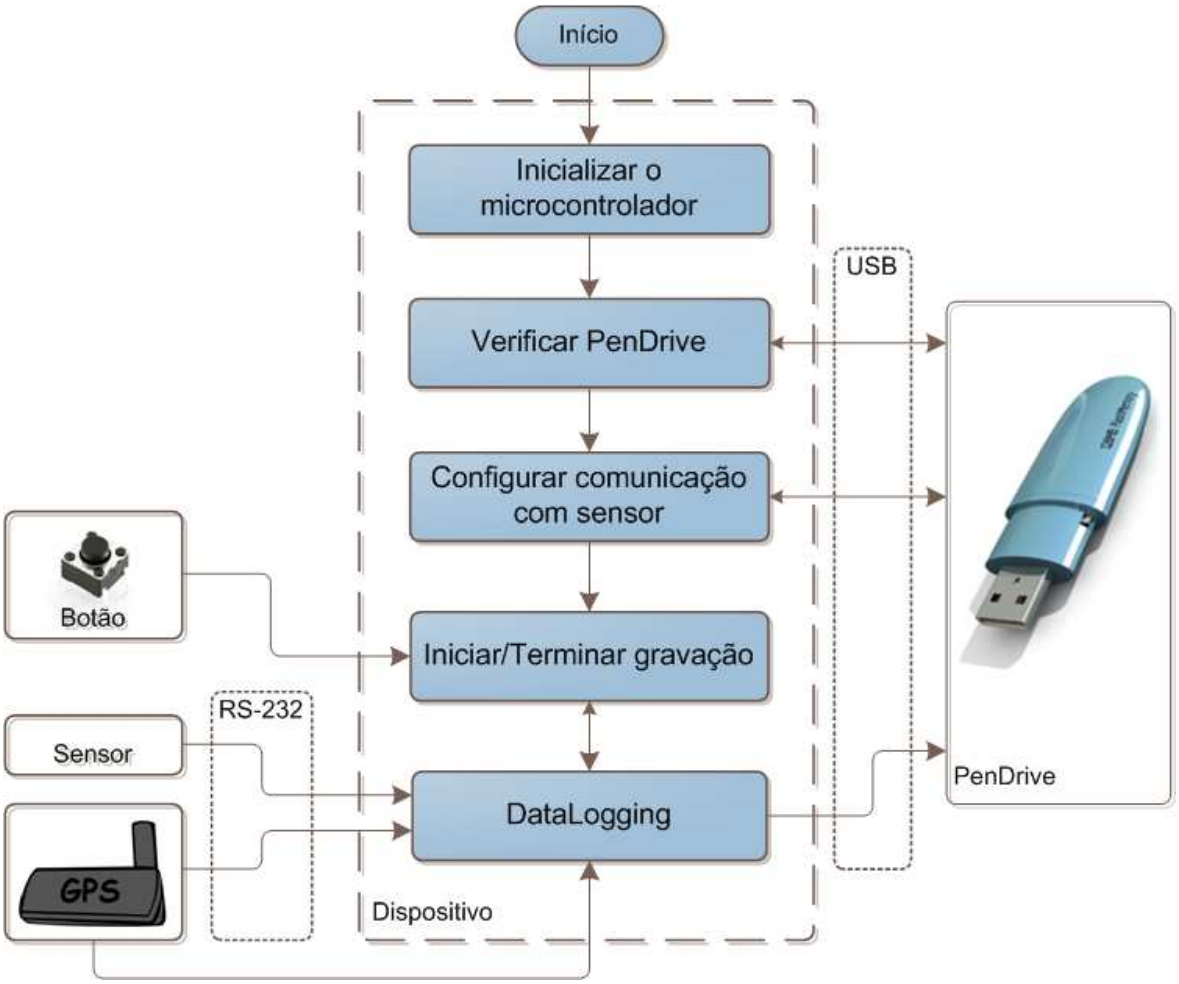


**Figura 21: Diagrama de blocos do *firmware* do DataLogger.**

O *firmware* desenvolvido, cujo diagrama de blocos está apresentado na figura anterior, é composto principalmente por cinco fases: Inicialização do micro-controlador PIC32MX440F512H, detectar a existência de uma PenDrive USB, configurar a comunicação com o sensor, esperar a ordem de início da gravação e a fase de recepção, processamento e registo de dados. Em qualquer momento da fase de gravação pode ser activado o botão. Desta forma é terminada a gravação e volta-se para a fase de espera. Finalmente será abordada a sincronização temporal.

### 4.3.1 Inicialização do micro-controlador

Assim que é ligado o micro-controlador, este começa por configurar as principais funcionalidades. Como podemos ver pela figura 22, é primeiramente feita a configuração dos pinos que controlam os LEDs de sinalização, sendo estes ligados imediatamente. De seguida, são configuradas as interrupções, suas prioridades e é activado o sistema de interrupções por vectores. Por fim é inicializada a *stack* USB.

No final da inicialização do micro-controlador são desactivados ambos os LEDs.

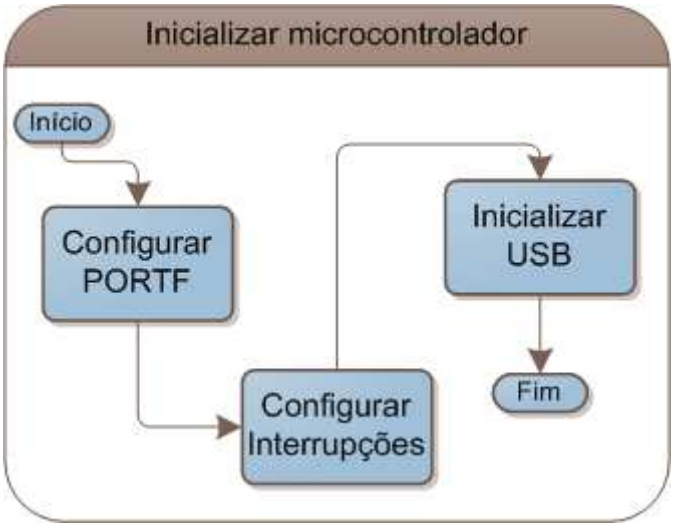


**Figura 22: Diagrama de blocos da inicialização do micro-controlador.**

### 4.3.2 Verificar PenDrive USB

Com as inicializações feitas, o passo que se segue é verificar a ligação com a PenDrive.

Como se pode ver pela figura 23, esta verificação inicia-se com a chamada da função USBTasks(), função esta que executa tarefas relacionadas com a operação do controlador USB do PIC32 como *host*. Tem como propósito principal o controlo da conexão/desconexão e enumeração do *device* em causa.

Posteriormente é verificado se o sistema de ficheiros definido no dispositivo PenDrive é compatível (apenas FAT16 ou FAT32).

Caso tudo esteja correcto é activado o LED 1 de forma a sinalizar que a PenDrive está pronta a ser utilizada (caso contrário este LED não se acenderá, indicando que a PenDrive inserida não é reconhecida ou tem uma formatação incompatível).

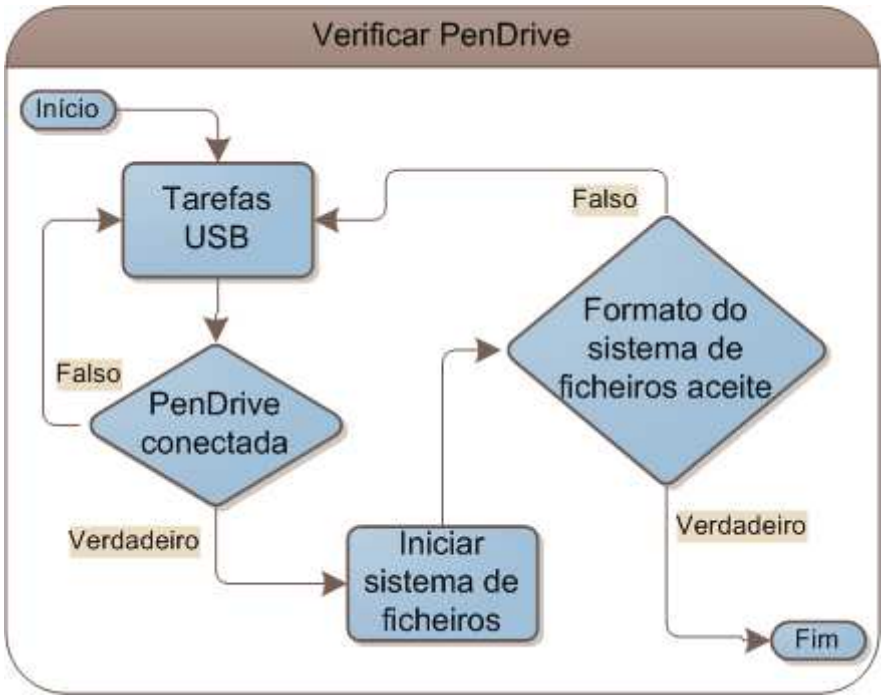


**Figura 23: Diagrama de blocos da verificação da PenDrive.**

### 4.3.3 Configurar a comunicação com sensor

De forma a melhorar o uso do DataLogger, a configuração da comunicação com o sensor é algo que deve ser possível de forma simples. Assim sendo, existem duas formas de configurar esta comunicação: definir num ficheiro de configuração previamente criado na PenDrive ou, caso isso não seja possível, é aplicada uma configuração pré-definida. Este ficheiro de configuração contém os dados relativos ao protocolo série a implementar entre o PIC32 e o sensor, nomeadamente

especifica o *baudrate*, a paridade, o número de bits por palavra, e o número de *stop bits* de uma comunicação série via UART. Nesta fase, e em simultâneo, também é feita a configuração da comunicação com o receptor GPS.

Refere-se que, neste momento, não foram considerados outros protocolos de comunicação série que não o via UART. Isto deveu-se à necessidade mais imediata de armazenar os dados dos sensores utilizados no projecto de investigação INSHORE, e estes operam através deste protocolo, pelo que se deu mais importância à comunicação via UART. Contudo, modificações muito simples ao protótipo desenvolvido permitem a implementação de canais de comunicação baseados em diferentes protocolos série (uma vês que o PIC32 dispõe, já, de diferentes tipos de interface série, requerendo apenas uma ligação directa aos sensores, ou pouco mais).

Como se pode ver pela figura 24, quando a PenDrive é validada pelo micro-controlador são criadas directorias (caso estas não existam) apenas para informação referente ao DataLogger. Nesta fase é também feita a configuração com o receptor GPS

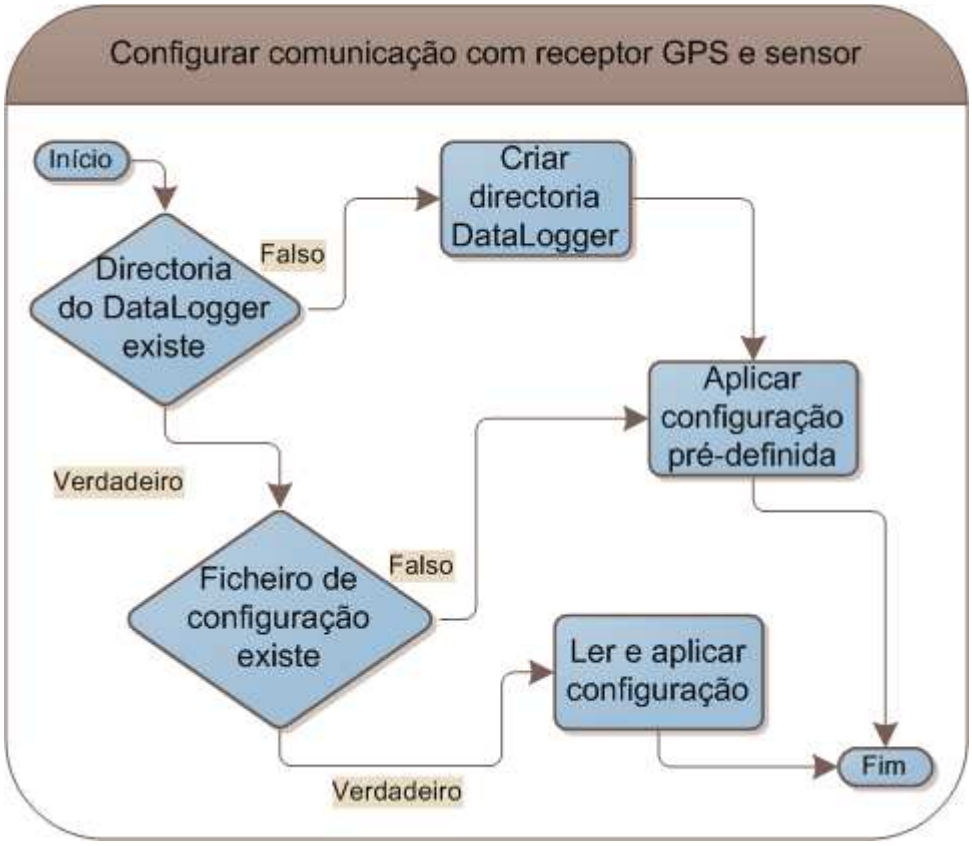


**Figura 24: Diagrama de blocos de configuração das comunicações.**

### 4.3.4 Recepção, processamento e registo de dados

Até aqui o funcionamento descrito apenas incide sobre a preparação do micro-controlador, porém é nesta fase que o DataLogger realiza o papel desejado.

Após toda a configuração e verificação, o micro-controlador activa o LED 0 para que o utilizador saiba que pode iniciar o registo dos dados. Ao pressionar o botão (o que causa a activação de uma linha de interrupção no PIC32), é iniciado o processo de DataLogging.

Como se pode ver pela figura 25, o micro-controlador cria e abre em modo de escrita um novo ficheiro de registo. O DataLogger recebe uma mensagem do sensor, processa-a, sincroniza o tempo e regista a informação na PenDrive. Finalmente, verifica a *flag* que indica o início ou a paragem do processo de registo e age em conformidade. Durante este processo, o LED 0 está sucessivamente a

alternar para que o utilizador saiba que o DataLogger se encontra no modo de armazenamento de dados. Quando acaba a gravação (por comando do utilizador através do botão), o micro-controlador desactiva o LED 0 (apenas voltando a activar o mesmo quando estiver pronto para a iniciar a gravação) e termina o ficheiro de registo. O ciclo efectuado pelo DataLogger enquanto procede ao registo de dados é apresentado na figura 25. Como se pode verificar, depois de o DataLogger verificar e configurar o hardware ao qual está ligado, ele permanece no modo de funcionamento descrito na figura 25 ou no estado de espera de início da gravação.

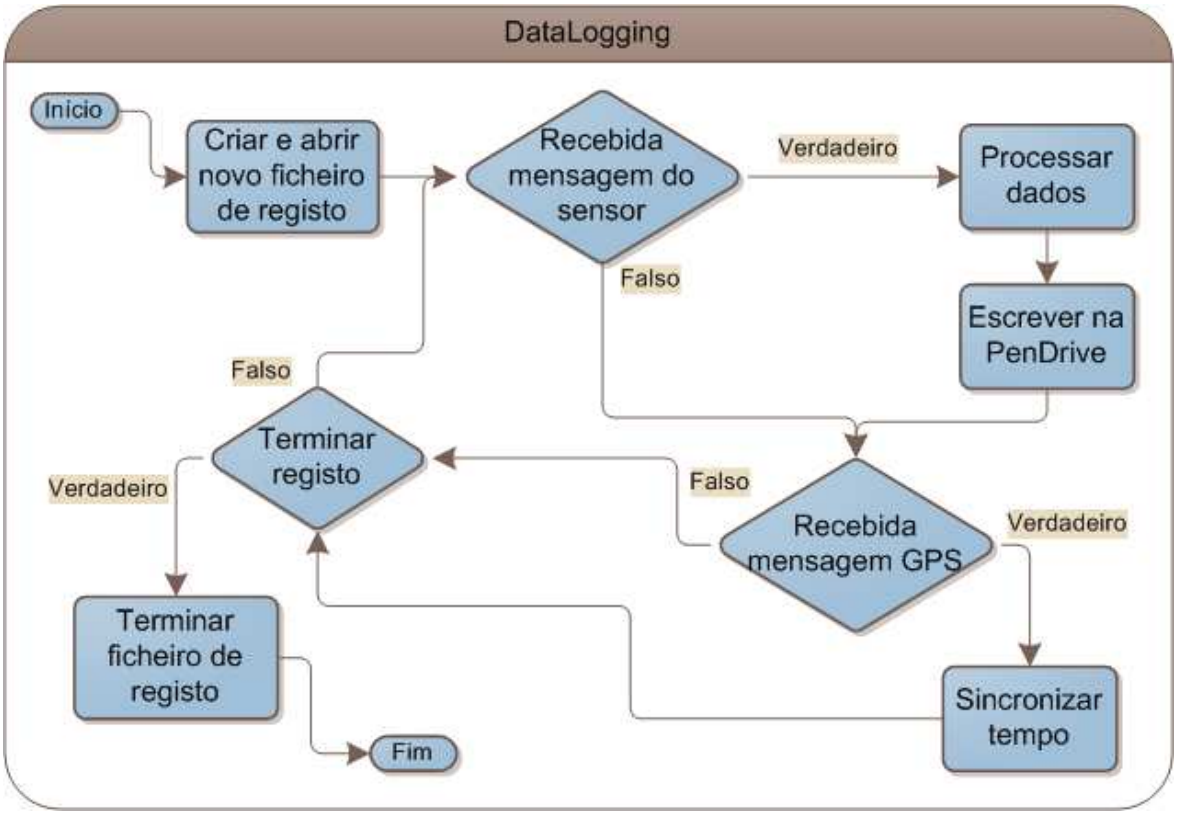


**Figura 25: Diagrama de blocos DataLogging.**

A recepção de dados do sensor é efectuado através de um evento servido por interrupção no PIC32, pois o método de comunicação utilizado é *null modem* e não existem sinais de controlo na comunicação. Assim, cada byte que é enviado ao PIC32 é sinalizado por uma interrupção, cuja rotina respectiva se encarrega de

registar o byte recebido, libertando a UART para a recepção do próximo byte. Dependendo da cadência de chegada da informação enviada pelo sensor, o DataLogger acumula um conjunto de bytes recebidos, encapsula esses dados numa trama devidamente definida (a especificação do formato destas tramas é apresentada na Secção 4.4), à qual é adicionado o instante de tempo em que esses dados foram recolhidos (com base no relógio interno do PIC32 que foi criado para o efeito e que é descrito na secção seguinte), sendo essa trama armazenada no ficheiro respectivo na PenDrive.

### 4.3.5 Processo de sincronização temporal

O processo de sincronização temporal dos dados enviados pelo sensor é um ponto fundamental deste trabalho. Este proporciona ao DataLogger uma característica que o distingue de outros dispositivos existentes no mercado, uma vez que permite que diferentes DataLoggers colocados em pontos geográficos diferentes (logo, ligados a sensores diferentes) armazenem os dados dos sensores de uma forma síncrona, podendo esses dados ser correlacionados temporalmente de uma forma directa e inequívoca. Para proceder a esta sincronização o DataLogger necessita ter acesso a um relógio de tempo universal (ou, pelo menos, a um sinal de sincronismo que seja comum a outros DataLoggers). Para a realização deste trabalho considerou-se a base de tempo gerada por um receptor GPS de muito baixo custo.

Para além da posição geográfica, um receptor GPS determina, também, o valor do tempo em que se situa com uma precisão de cerca de 50 ns relativamente ao sistema de tempo GPS (que é mantido por vários osciladores de rubídio e césio colocados em algumas estações terrestres situadas em vários pontos do globo terrestre). Assim, todos os receptores GPS que captem os sinais enviados pelos satélites GPS ficam imediatamente sincronizados entre si (a menos de um erro de cerca de 50 ns).

Alguns modelos de receptor GPS disponibilizam para o exterior um sinal lógico (normalmente compatível com os níveis TTL) que gera um impulso cuja transição ascendente é síncrona com o instante de início de cada segundo do sistema de tempo GPS. Este sinal tem a denominação genérica de 1PPS por gerar um pulso por segundo.

Assim, foi utilizado neste trabalho um destes receptores (modelo LassenSQ do fabricante *Trimble*) que disponibiliza o sinal 1PPS, sendo este sinal aplicado directamente a uma linha de interrupção do PIC32. A rotina de atendimento a esta interrupção é responsável pelo acerto do relógio interno do PIC32 que fora propositadamente criado para o efeito. Este relógio consiste num registo de memória de 4 bytes que é incrementado a cada milissegundo por uma rotina de atendimento a um *timer* interno do PIC32, programado para disparar a cada milissegundo. Quando ocorre uma interrupção do 1PPS, a respectiva rotina simplesmente arredonda para o segundo exacto o valor contido no registo do relógio do PIC32. Ou seja, embora no intervalo de um segundo o tempo evolua com base no oscilador do PIC32, é garantido que essa contagem comece sempre de zero em cada segundo, evitando assim erros significativos do relógio do PIC32 pelo facto de o seu oscilador não constituir uma referência temporal universal. Mais ainda, esta implementação simples é, também, muito robusta pois, se por algum motivo o sinal 1PPS não for enviado (porque o receptor GPS ficou momentaneamente sem conseguir captar os sinais dos satélites GPS, ou por um outro qualquer motivo), o relógio interno do PIC32 continua a ser actualizado pelo *timer* do PIC32 (embora, com o passar do tempo, este relógio apresentará um desvio relativamente ao sistema de tempo GPS).

Contudo, o sistema de sincronização apresentado no parágrafo anterior apenas sincroniza o tempo relativo do relógio interno do PIC32, e não o seu valor absoluto. Ora, este acerto é fundamental para permitir o sincronismo de diferentes DataLoggers. Para efectuar o acerto do valor absoluto do relógio do PIC32 são recebidos, também, os dados que o mesmo receptor GPS envia pela sua porta série (via UART). Nos dados enviados pelo LassenSQ está não só a posição geográfica

da sua antena mas também o instante de tempo em que essa posição é calculada, sendo este cálculo efectuado a cada segundo exacto do tempo GPS. Assim, depois de receber o sinal 1PPS, o DataLogger fica atento às mensagens enviadas pela porta série do receptor GPS, descodificando a mensagem que contém a informação temporal da última posição calculada. O valor do tempo contido nesta mensagem corresponde ao instante de ocorrência do último sinal 1PPS, logo o DataLogger actualiza o valor do relógio interno do PIC32 com base neste valor. Assim, o tempo interno do PIC32 fica síncrono com o tempo universal GPS. De notar que o tempo GPS corresponde ao valor total de segundos que decorreram desde a última transição de Sábado para Domingo, sendo este valor reinicializado todas as semanas. O valor do registo que é mantido no PIC32 relativo ao seu relógio interno também segue esta definição de tempo.

Quando um conjunto de dados recebidos do sensor é processado para ser armazenado na PenDrive, estes dados são encapsulados numa trama sendo um dos campos do cabeçalho o valor do registo do relógio interno do PIC32 nesse instante. Desta forma os dados armazenados na PenDrive ficam associados a uma etiqueta temporal síncrona com o sistema de tempo GPS.

## 4.4 Especificação do Conteúdo dos Ficheiros do DataLogger

A criação de ficheiros de registo foi determinada logo na fase de idealização dos objectivos do projecto. Contudo, com o decorrer do desenvolvimento, sentiu-se a necessidade de configurar a comunicação entre o DataLogger e o sensor, colocando um ficheiro de configuração na respectiva PenDrive.

Sendo RS-232 o protocolo utilizado, definições como *baudrate*, *data bits*, *stop bits* e paridade podem variar entre sensores do mesmo género. De forma a criar um DataLogger mais versátil, esta configuração teria de ser executada de um modo eficiente. Reprogramar o micro-controlador sempre que se mudar para um

sensor com diferentes definições de comunicação é algo impraticável. No entanto, implementar um ficheiro de configuração na PenDrive é algo simples, prático e eficaz. Uma vez que este ficheiro de configuração é editável directamente num computador (tratando-se de um ficheiro de texto), o utilizador pode, assim, de uma forma muito simples, adaptar o DataLogger a sensores com configurações específicas para a comunicação série.

O referido ficheiro de configuração, de nome "Config.txt", deve estar presente no directório ".\DATALOG\CONFIG " sendo o formato do seu conteúdo apresentado na figura 26.

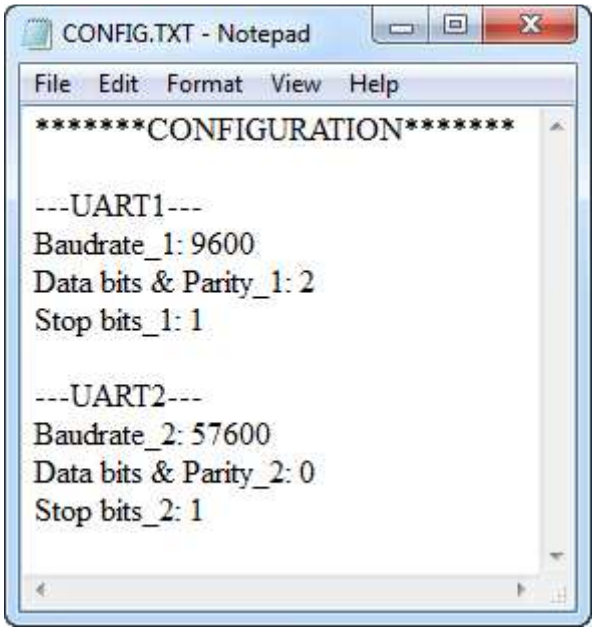


**Figura 26: Ficheiro de configuração.**

Como se pode ver pela figura 26, podem ser configuradas ambas as UARTs. A configuração da UART1 é relativa à comunicação com o receptor GPS e a configuração da UART2 diz respeito à comunicação com o sensor. Os possíveis valores e configurações são especificados na tabela 2.

| Opções do ficheiro de configuração | |
|---|---|
| ***Baudrate*** | De 306 a 20.000.000 |
| ***Databits & Parity*** | 0 (8-bit data, sem paridade) |
| | 1 (8-bit data, paridade par) |
| | 2 (8-bit data, paridade impar) |
| | 3 (9-bit data, sem paridade) |
| ***Stop bits*** | 1 |
| | 2 |

**Tabela 2: Opções do ficheiro de configuração.**

Os ficheiros de registo que armazenam os dados recolhidos do sensor estão presentes no directório ".\DATALOG\DATA\" da PenDrive e são numerados de forma crescente entre "LOG001.txt" e "LOG999.txt". Como referido anteriormente, os dados recebidos são colocados dentro de uma trama de bytes com as seguintes características:

| Carácter inicial | Comprimento dos dados | Etiqueta temporal | Dados | Checksum | Carácter terminador |
|---|---|---|---|---|---|

**Figura 27: Trama de mensagens gravadas na PenDrive.**

| Campo | Características |
|---|---|
| Carácter inicial | 0xAA (1 byte) |
| Comprimento dos dados | Número de bytes do campo de dados (1byte) |
| Etiqueta temporal | Tempo em milissegundos (inteiro de 4 bytes) |
| Dados | Dados do sensor (min: 1 byte; max: 248 bytes) |
| Checksum | XOR aos bytes anteriores excepto inicial |
| Carácter terminador | 0x55 |

**Tabela 3: Características da trama armazenada.**

Como se pode ver pela figura 27, a trama é composta por seis campos e através da tabela 3, é possível verificar as características de cada campo. Os

caracteres inicial e terminador indicam respectivamente, o início e o fim da mensagem. O "comprimento dos dados" é 1 byte que indica o comprimento total do campo de dados em bytes. "Etiqueta temporal" é referente a um inteiro de 4 bytes que contém o tempo (em ms) do início da mensagem. O campo de "dados" tem um comprimento máximo de 248 bytes e mínimo de 1 byte. Finalmente, o campo "checksum" tem o intuito de detectar erros. O seu valor é determinado aplicando a operação XOR aos bytes dos campos "comprimento dos dados", "etiqueta temporal" e "dados". Cada trama é, então, registada sequencialmente no ficheiro de registo criado na PenDrive para o efeito.

# Capítulo 5

# 5 Teste do DataLogger

Neste capítulo são descritos vários testes que verificam o bom funcionamento do DataLogger desenvolvido.

Inicialmente, realizaram-se vários testes ao PIC32 *kit board*, de forma a verificar o seu bom funcionamento.

Apresenta-se o primeiro teste ao DataLogger, em que o receptor GPS é considerado o sensor.

De seguida, apresentam-se dois testes que verificam o funcionamento do DataLogger. Nestes testes, o sensor é simulado pelo computador e os dados são enviados de forma isolada ou em bloco.

Finalmente, é apresentado um teste do consumo do DataLogger em funcionamento.

## 5.1 Teste PIC32 *kit board*

Vários foram os testes realizados ao PIC32 *kit board*. Desde o funcionamento dos portos I\O, interrupções (internas e externas), comunicação RS-232, entre outros, todos foram realizados com sucesso.

O único teste de valor a referir, é o efectuado para averiguar o *clock* do PIC32 e a sua capacidade de executar uma instrução por ciclo de relógio.

Para verificar os 80MHz de *clock*, utilizou-se um osciloscópio digital com uma largura de banda de 500MHz e uma frequência de amostragem de 5 GSPS. Inicialmente programou-se o pino desejado como *output* e ligou-se o osciloscópio para medir o sinal presente. Programou-se o *clock* do PIC32 para funcionar a

80MHz e alternou-se o pino sucessivamente. Dado que, o código implementado consiste num ciclo *while* contendo 4 instruções para comutar o pino I\O, espera-se que uma em cada quatro comutações, uma seja mais demorada. Facilmente isto é verificado pela figura 28, em que temos um nível lógico mais longo a cada 4 comutações.

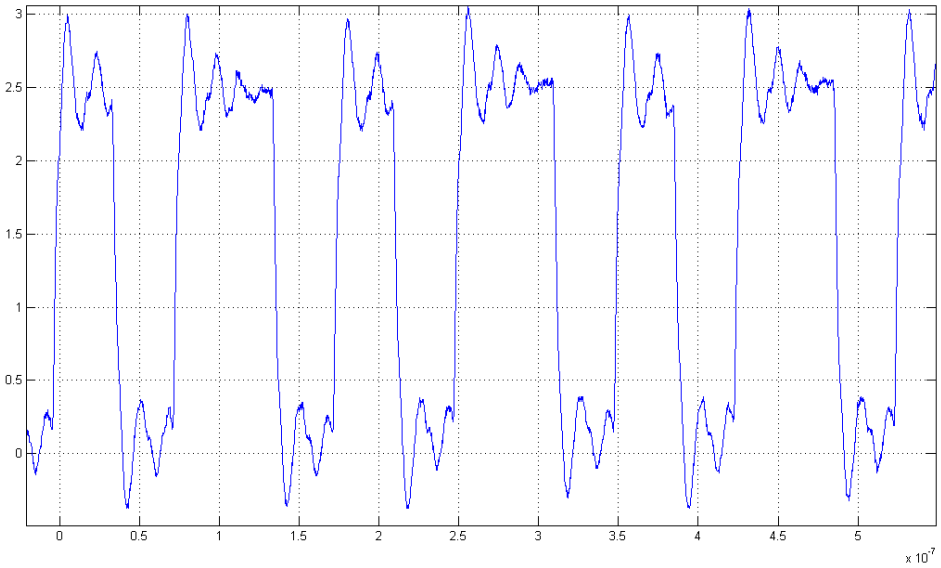


**Figura 28: Sinal de saída do pino I\O.**

Não considerando o estado *high* mais longo (devido a conter o tempo da instrução *while*), verifica-se que o tempo que demora cada transição é de cerca de $0.38*10^{-7}$ segundos. Apenas falta referir que, cada instrução de inversão do pino I\O em linguagem C, corresponde a três instruções em *assembly*. Assim, obtém-se $(0.38*10^{-7})\ /3 = 12.67*10^{-9}$ segundos por instrução em *assembly* o que corresponde a uma frequência de 78.9MHz.

Conclui-se, assim, que a programação do PIC32 está como era esperado pois a frequência de funcionamento obtida é muito próxima dos 80MHz.

## 5.2 Funcionamento apenas com um receptor GPS

Este teste consiste em ligar o sensor GPS através da comunicação série com o micro-controlador e, por intermédio deste sensor, fazer o registo dos dados por ele recebidos ao mesmo tempo que se procede à sincronização do tempo.

Pela figura 29 pode-se verificar as várias mensagens registadas pelo DataLogger, com os respectivos dados.

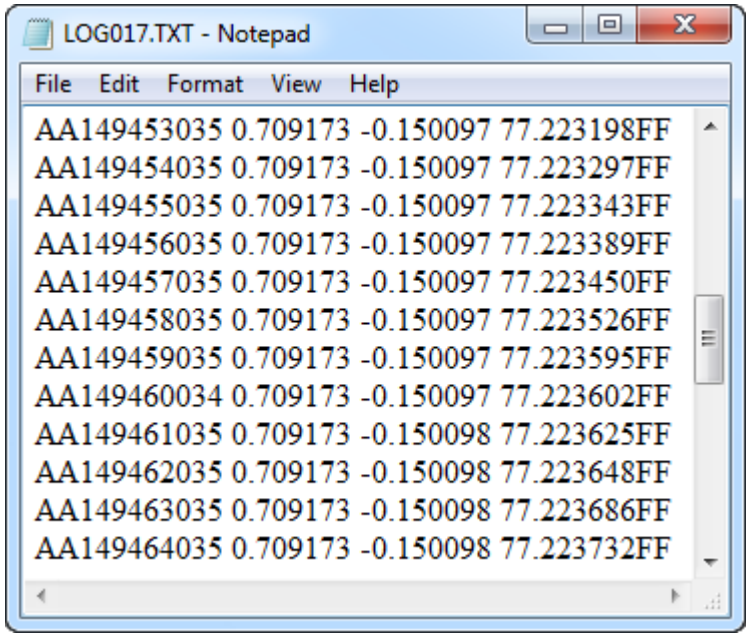


**Figura 29: Excerto de um ficheiro de registo.**

Sendo este um teste inicial, para verificar a gravação de dados e actualização do tempo recebido pelo GPS, foi implementado o tipo de mensagens da figura 30. Para facilitar a visualização dos dados, registou-se a informação desta experiencia em formato de texto.

| Caracteres Iniciais | Tempo do PIC32 em milissegundos | Dados do sensor | Caracteres Terminadores |
|---|---|---|---|

**Figura 30: Formato das mensagens para o teste do GPS.**

Fazendo uma análise à primeira linha da figura 29, pode-se verificar o seguinte:

- Caracteres início de trama: AA;
- Tempo do instante de recepção dos dados provenientes do sensor GPS, em milissegundos: 149453035;
- Dados enviados pelo sensor: 0.709173 -0.150097 77.223198;
- Caracteres de fim de trama: FF.

Convertendo o tempo da mensagem, verifica-se que corresponde a 1 dia, 17 horas, 30 minutos, 53 segundos e 35 milissegundos depois de uma meia-noite de Sábado para Domingo (ou seja, 17h30m53.35s de Segunda-feira).

Os três valores que aparecem no campo de dados do sensor representam respectivamente a latitude, longitude e altitude da posição calculada pelo receptor GPS que está a tomar o lugar de sensor alvo. Dado que os dois primeiros campos estão em radianos, obtém-se o seguinte:

- Latitude: 40.63261984 (graus)
- Longitude: -8.59992461 (graus)
- Altitude: 77.223198 (m)

Com estes valores de latitude e longitude confirma-se que o sensor está na região de Aveiro, exactamente (a menos de uns poucos metros) no local onde a experiência decorrera. O valor da altitude é referente a um elipsóide, de nome WGS84, que se aproxima da forma do globo terrestre. Este apresenta naturalmente erros relativamente à altura local medida relativamente ao nível médio das águas do mar.

## 5.3 Funcionamento com sensor

Os dois testes que se apresentam de seguida têm como intuito demonstrar o bom funcionamento do protótipo do DataLogger. Desta forma, tanto o receptor GPS, como o sensor estão ligados através do protocolo RS-232. Nestes testes, o sensor foi simulado por um computador.

Com estes testes, é possível verificar o funcionamento do DataLogger perante um sensor que envie dados esporadicamente ou, perante um sensor que envie dados a uma cadência elevada.

### 5.3.1 Envio de bytes isolados

Este teste realizado ao protótipo do DataLogger, consistiu em enviar caracteres isolados. Desta forma, foi utilizado o programa *HyperTerminal* para simular um sensor. Este foi configurado da seguinte forma:

- Baudrate: 57600;
- Data bits: 8;
- Paridade: Nenhuma;
- Stop bits: 1;
- Controlo da comunicação: Nenhum.

Dado que a informação contida nos ficheiros LOGxxx.TXT está em formato binário, não é possível verificar correctamente o seu conteúdo abrindo o ficheiro com um editor de texto (tal como na experiencia anterior). Assim, foi criado um pequeno programa em Linguagem C para realizar a visualização correcta dos ficheiros de gravação.

O programa criado lê o ficheiro de gravação e de seguia imprime as características: número de trama, etiqueta temporal e, finalmente, a sequência de bytes da trama.

```
trama01: 579792191 || AA_01_3F_ED_8E_22_61_1E_55_
trama02: 579792567 || AA_01_B7_EE_8E_22_61_95_55_
trama03: 579792951 || AA_01_37_F0_8E_22_61_0B_55_
trama04: 579793359 || AA_01_CF_F1_8E_22_61_F2_55_
trama05: 579798407 || AA_01_87_05_8F_22_63_4D_55_
trama06: 579798895 || AA_01_6F_07_8F_22_63_A7_55_
```

**Figura 31: Visualização do ficheiro de gravação do teste de envio de bytes isolados.**

Na figura 31 pode-se verificar que, nesta experiência, foram gravadas seis tramas com dados. Convertendo a etiqueta temporal da "trama01", verifica-se que corresponde às 17h3m12.191s de um Sábado. Respectivamente à "trama06", é possível constatar que foi registada 6 segundos e 704 milissegundos depois da "trama01". Verifica-se também que todas as tramas estão registadas no formato definido no subcapítulo 4.4.

Ao analisar o segundo byte (comprimento do campo de dados) de cada trama, confere-se que qualquer uma delas apenas tem um byte no campo de dados. Este comportamento é desejado para que a etiqueta temporal associada aos dados recebidos seja a mais exacta possível.

Apenas falta referir que os dados enviados pelo computador foram quatro caracteres 'a' e dois 'c'.

### 5.3.2 Envio de um bloco de bytes

Depois de testar com sucesso o comportamento do DataLogger com sensores que apenas enviem bytes isolados, faltava ainda testar o seu comportamento com sensores que enviem blocos de bytes sem interrupções. Assim, usando o mesmo programa e as mesmas configurações do teste anterior, foram enviados inicialmente alguns caracteres isolados e de seguida um ficheiro de texto.

Para esta experiência, o programa criado para visualizar o ficheiro de gravação foi alterado, introduzindo, depois da trama, a confirmação do campo *checksum*.

```
trama01: 596742239 || AA_01_5F_90_91_23_41_3D_55_ || 3D

trama02: 596742736 || AA_01_50_92_91_23_41_30_55_ || 30

trama03: 596743064 || AA_01_98_93_91_23_41_F9_55_ || F9

trama04: 596743367 || AA_01_C7_94_91_23_41_A1_55_ || A1

trama05: 596760580 || AA_8B_04_D8_91_23_49_73_74_6F_20_E9_20_75_6D_20_66_69_63_6
8_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_37_38_39_20_
46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64
_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_2E_2
0_49_73_74_6F_20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_
65_2E_20_30_31_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_1F_55_ || 1F

trama06: 596760617 || AA_F8_29_D8_91_23_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_6
9_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_37_38_
39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F
_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2
E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_
73_74_65_2E_20_30_31_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F
_20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_3
1_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_
66_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_37
_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_D7_55_ || D7

trama07: 596760664 || AA_F8_58_D8_91_23_69_63_68_65_69_72_6F_20_64_65_20_74_65_7
3_74_65_2E_20_30_31_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_
20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31
_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_3
7_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_55_55_49_73_74_
6F_20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30
_31_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_2
0_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_
37_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_69_63_68_65_69
_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_37_89_55_ || 3B

trama08: 596760792 || AA_F8_D8_D8_91_23_69_63_68_65_69_72_6F_20_64_65_20_74_65_7
3_74_65_2E_20_30_31_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_
20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31
_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_6
6_69_63_68_65_69_72_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_37_
38_39_20_46_49_4D_2E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_69_63_68_65_69_72
_6F_20_64_65_20_74_65_73_74_65_2E_20_30_31_32_33_34_35_36_37_38_39_20_46_49_4D_2
E_2E_2E_20_49_73_74_6F_20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64_65_20_74_
65_73_74_65_2E_20_30_31_32_33_34_35_36_37_38_39_20_46_49_4D_2E_2E_2E_20_49_73_74
_6F_20_E9_20_75_6D_20_66_69_63_68_65_69_72_6F_20_64_65_20_74_65_DD_55_ || DD
```

**Figura 32: Visualização do ficheiro de gravação do teste de envio de um bloco de bytes.**

Como se pode ver pela figura 32, foram enviados quatro caracteres e de seguida um ficheiro de texto. A etiqueta temporal da "trama05" corresponde às 21h46m0.580s de um Sábado e, as tramas seguintes, foram registadas alguns milissegundos depois.

Fazendo uma análise ao campo *checksum* e respectiva confirmação, verifica-se que, de todas as tramas, apenas a "trama07" contém dados corrompidos.

A indicar ainda que todos os dados enviados pelo computador foram registados no ficheiro de gravação.

Com este teste, verifica-se o bom funcionamento do DataLogger com sensores que enviem dados a um ritmo elevado.

## 5.4 Consumo energético do DataLogger

O teste realizado para determinar o consumo energético, consistiu em medir quer a tensão, quer a corrente que o DataLogger consome em funcionamento. Neste teste foi utilizado o multímetro digital de gama baixa.

Nas medições realizadas, os resultados obtidos foram os seguintes:

- Tensão: 12V;
- Corrente: 0.110A;

Com estes valores, determina-se que a potência do DataLogger é de 1.32W. Se este for alimentado por uma bateria de 12V - 3.5Ah (bateria utilizada no projecto de investigação INSHORE), então o DataLogger permanecerá em funcionamento durante um período de cerca de 32 horas.

Tendo em conta o tipo de aplicação, pode-se concluir que o DataLogger tem um baixo consumo energético, algo que é indispensável para a sua portabilidade.

# Capítulo 6

# 6 Conclusões e Trabalho Futuro

Neste capítulo são apresentadas as conclusões globais e algumas propostas de possível trabalho futuro.

## 6.1 Conclusões

O trabalho realizado incide na implementação de um DataLogger de filosofia simples, com tamanho reduzido, que permita um elevada capacidade de armazenamento de dados, com capacidade de se ligar a diversos tipos de sensores e com a possibilidade de este receber um sinal de sincronismo temporal.

O DataLogger implementado, descrito no capítulo 4, possui a capacidade de ligação a uma PenDrive USB genérica, proporcionando assim uma elevada capacidade de armazenamento com muita flexibilidade na transferência dos dados armazenados para um computador, tem a possibilidade de comunicar com o sensor através do protocolo série RS-232, recebe um sinal de sincronismo (e consequentemente a informação temporal) proveniente de um receptor GPS (ou de outro qualquer dispositivo cujo objectivo é criar uma referência temporal). Devido ao sistema de ficheiros da PenDrive e do formato dos próprios ficheiros, estes são directamente reconhecíveis por qualquer sistema operativo, tornando acessível a sua leitura de uma forma muito simples. O protótipo do DataLogger é de dimensões reduzidas e, devido aos componentes utilizados, este consome pouca energia. Antevendo a possibilidade da sua utilização em diversos ambientes, o DataLogger foi propositadamente projectado para poder ser alimentado em qualquer valor da gama de +6.5V a +32V.

O DataLogger tem ainda a capacidade de ler um ficheiro de configuração da comunicação a partir da PenDrive USB.

A capacidade de ser um elemento SyncMaster ou SyncSlave, inicialmente pretendida para a ligação directa entre dois, ou mais, DataLoggers, para que todos se sincronizassem a uma só referência de tempo, não foi implementada por se ter verificado ser mais simples a adição de um receptor GPS de baixo custo (e muito pequenas dimensões) a cada novo DataLogger a utilizar, tornando assim todos os DataLoggers independentes uns dos outros, contudo mantendo o sincronismo temporal entre eles.

Entende-se, assim, que o DataLogger desenvolvido preenche os principais requisitos inicialmente propostos, sendo um dispositivo muito versátil para o armazenamento de dados de vários sensores, armazenamento este efectuado com sincronismo temporal entre vários DataLoggers.

## 6.2 Trabalho Futuro

Quanto aos objectivos que ficaram por realizar, pensa-se que a sua implementação seja possível e de muita utilidade.

Relativamente ao trabalho realizado, deixam-se algumas sugestões de trabalho futuro.

- Passar de protótipo do DataLogger para o formato de produto final fazendo uma implementação em placa de circuito impresso (PCB) com elementos SMD para assim se reduzir ainda mais as suas dimensões, sendo inclusivamente colocado dentro de uma caixa apropriada.
- Incluir, também, uma interface simples de hardware que permita seleccionar entre os protocolos de comunicação série disponibilizados pelo PIC32 (I2C, SPI) e outros (por exemplo, RS485). Sugere-se que esta selecção seja feita, também, através do ficheiro de configurações armazenado na PenDrive, permitindo ao PIC32 configurar-se adequadamente (e, eventualmente, seleccionar as ligações de hardware convenientes à porta de ligação com o sensor).

# Bibliografia


1. Wikipedia, http://en.wikipedia.org/wiki/Data_logger. Novembro de 2010.
2. Keskull, R.I., http://homepages.tig.com.au/~robk/logger-serial.html. Novembro de 2010.
3. Brookhouse, http://brookhouseonline.com/nmealogger.htm. Outubro de 2010.
4. MicroDAQ, http://www.microdaq.com/datataker/15-analog-input-channels.php. Outubro de 2010.
5. AGGsoftware, http://www.aggsoft.com/nmea-data-logger.htm. Outubro de 2010.
6. Neal, S., http://www.electronicinterfaceprojects.com/. Novembro 2010.
7. Pallas-Areny, F.V.-P.a.R., *Microcontrollers Fundamentals and Applications with PIC*. 2009, CRC.
8. L. D. Jasio, T.W., D. Ibrahim, J. Morton, M. P. Bates, J. Smith, D. W. Smith, and C. Hellebuyck, *PIC Microcontrollers (Newnes Know It All)*. 2007, Boston: Newnes.
9. Microchip, http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2018&mcparam=en532888. Outubro de 2010.
10. Microchip, http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en535591. Outubro de 2010.
11. *PIC32MX3XX/4XX Family - DS61143E*. 2008, Data Sheet: Microchip Technology Inc.
12. Microchip, http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2519&param=en534451&page=wwwdevMPLABEmulatorDebuggers. Outubro de 2010.
13. Cadence. http://www.cadence.com/products/pcb/Pages/default.aspx. Outubro 2010.

# Anexo A – *Pinout* do PIC32 *kit board*

| Pinos PIC | Pinout | | Pinos PIC | Pinos PIC | Pinout | | Pinos PIC |
|---|---|---|---|---|---|---|---|
| D+/RG2 - 37 | 15 | 14 | 42 -IC1/RTCC/INT1/RD8 | VUSB – 35 | 14 | 15 | 36 - D-/RG3 |
| IC2/U1CTS//INT2/SDA1/RD9- 43 | 16 | 13 | 44 -IC3/PMCS2/PMA15/INT3/SCL1/RD10 | USBID/RF3 - 33 | 13 | 16 | 34 – VBUS |
| IC4/PMCS1/PMA14/INT4/RD11- 45 | 17 | 12 | 46 -OC1/INT0/RD0 | PMA9/U2RX/SDA2/CN17/RF4 – 31 | 12 | 17 | 32 - PMA8/U2TX/SCL2/CN18/RF5 |
| OC2/U1RTS/BCLK1/RD1- 49 | 18 | 11 | 50 -OC3/U1RX/RD2 | PMALH/PMA1/U2RTS/BCLK2/AN14/RB14 – 29 | 11 | 18 | 30 - PMALL/PMA0/AN15/OCFB/CN12/RB15 |
| OC4/U1TX/RD3- 51 | 19 | 10 | 52 -PMWR/OC5/IC5/CN13/RD4 | TCK/PMA11/AN12/RB12 – 27 | 10 | 19 | 28 - TDI/PMA10/AN13/RB13 |
| PMRD/CN14/RD5 - 53 | 20 | 9 | 54 - CN15/RD6 | VSS – 9,25,41 | 9 | 20 | 10,26,38 – VDD |
| CN16/RD7- 55 | 21 | 8 | 56 -VCAP/VDDCORE | TDO/PMA12/AN11/RB11 – 24 | 8 | 21 | 23 - TMS/CVREFOUT/PMA13/AN10/RB10 |
| RF0- 58 | 22 | 7 | 57 –ENVREG | PMA7/C2OUT/AN9/RB9 – 22 | 7 | 22 | 21- U2CTS/C1OUT/AN8/RB8 |
| PMD0/RE0- 60 | 23 | 6 | 59 -RF1 | AVSS - 20 | 6 | 23 | 19 – AVDD |
| PMD2/RE2- 62 | 24 | 5 | 61 -PMD1/RE1 | PGED2/AN7/RB7 – 18 | 5 | 24 | 17 - PGEC2/AN6/OCFA/RB6 |
| PMD4/RE4 - 64 | 25 | 4 | 63 -PMD3/RE3 | PGC1/EMUC1/AN1/VREF-/CVREF-/CN3/RB1 – 15 | 4 | 25 | 16 - PGD1/EMUD1/PMA6/AN0/VREF+/CVREF+/CN2/RB0 |
| PMD6/RE6 - 2 | 26 | 3 | 1 - PMD5/RE5 | C2IN-/AN2/CN4/RB2 – 14 | 3 | 26 | 13 - C2IN+/AN3/CN5/RB3 |
| PMA5/SCK2/CN8/RG6 - 4 | 27 | 2 | 3 - PMD7/RE7 | C1IN-/AN4/CN6/RB4 – 12 | 2 | 27 | 11 - VBUSON/C1IN+/AN5/CN7/RB5 |
| PMA3/SDO2/CN10/RG8 - 6 | 28 | 1 | 5 -PMA4/SDI2/CN9/RG7 | PMA2/SS2/CN11/RG9 - 8 | 1 | 28 | 7 – MCLR |

| Programador | |
|---|---|
| Pino | Nome/Função |
| 1 | VPP MCRL |
| 2 | VDD |
| 3 | VSS |
| 4 | ICSPDAT/PGD |
| 5 | ICSPCLK/PGC |
| 6 | Not Connected |

# Anexo B – Camada Silkscreen do PIC32 *kit board*